修 士 論 文

エージェント間の通信チャネルの変化に対応した動作機構の構築

北陸先端科学技術大学院大学
情報科学研究科情報科学専攻

廣瀬 伸也

2014 年 3 月

修 士 論 文

エージェント間の通信チャネルの変化に対応した動作機構の構築

指導教官　東条 敏 教授

審査委員主査　東条 敏 教授
審査委員　飯田 弘之教授
審査委員　島津 明 教授

北陸先端科学技術大学院大学
情報科学研究科情報科学専攻

1110051 廣瀬 伸也

提出年月: 2014 年 2 月

概 要

エージェントとは分散人工知能の分野では「知的なソフトウェア」の代名詞として使われている．マルチエージェントシステムは複数のエージェントが相互作用をしながら問題の解決を行うシステムであり, エージェント単体では難しい問題でも解決が可能となる．エージェント間のつながりを通信チャネルとして論理体系に導入する研究が行われている．エージェント間通信に関する先行研究としてはFIPA, 小林, Benthem らによるものがある．FIPA はエージェント間のコミュニケーションの形式化を行い, 小林らは FIPA でのエージェント間通信の成否を通信チャネルと関連付けて形式化する論理体系 BUL を提案した．Benthem らは他のエージェントがどういう状態かを問い合わせる通信を扱っている．一方でエージェントの推論パターンの先行研究としては Seligman らのものがある．Seligman らはコミュニティにおけるエージェント間の知識・信念・選好の動的変化を捉える研究プログラムを提案している．具体例として SNS の Facebook で考え, その中でも知識の課題に関しては Facebook Logic という新しい論理を提案している．Facebook Logic は従来の知識・信念の論理とは違い, エージェントの集団にエージェント同士がどのような関係性があるかを反映することができる．Facebook Logic の Facebook での友人関係をエージェントのチャネルの有無と読み替えることでエージェント間通信にも応用できる．通信によるエージェントの変化については Pimolluck らが commitment の告知と permission の告知を定義しており, これらの告知はエージェントに「∼ せよ」や「∼ してもよい」といった情報を送ることで, 受信側のエージェントの世界間のアクセス関係を削除や追加を行うことが可能である．commitment の告知については佐野ら [3] によってチャネル通信を扱うことができるように定義されているが, permission の告知はチャネルを用いた定義されておらずチャネルの変化に対応することができていない．

本研究では, チャネル通信を扱える commitment の告知を元にチャネル通信に対応した permission の告知を定義する．そして現実問題で考えたとき, 故障などにより通信チャネルが使えない場合も考えられる．そこで告知による通信チャネルの追加・削除を扱う論理を定義し, エージェント間通信の変化にも対応したシステムを構築, 検証を行う．エージェントの関係性の表現に論理 BUL 用いて, エージェント間通信と通信チャネルの有無の確認通信を形式化する．エージェント間通信については対象のエージェントのチャネルが存在しない場合に, 他のエージェントに通信対象エージェントとの通信チャネルの有無を問い合わせ, もしチャネルが確認できた場合に告知により通信対象のエージェントと間接的な通信を行い, チャネルが確認できない場合は切断されたエージェントの行動を予測して協調行動させる．

目 次

第1章　はじめに

1.1　研究の背景

エージェントとは分散人工知能の分野では「知的なソフトウェア」の代名詞として使われている. マルチエージェントシステムは複数のエージェントが相互作用をしながら問題の解決を行うシステムであり, エージェント単体では難しい問題でも解決が可能となる. エージェント間のつながりを通信チャネルとして論理体系に導入する研究が行われている. エージェント間通信に関する先行研究としてはFIPA [2], 小林 [4], Benthem [1] らによるものがある. FIPA はエージェント間のコミュニケーションの形式化を行い, 小林らは FIPA でのエージェント間通信の成否を通信チャネルと関連付けて形式化する論理体系 BUL を提案した. Benthem らは他のエージェントがどういう状態かを問い合わせる通信を扱っている. 一方でエージェントの推論パターンの先行研究としては Seligman [8] らのものがある. Seligman らはコミュニティにおけるエージェント間の知識・信念・選好の動的変化を捉える研究プログラムを提案している. 具体例として SNS の Facebook で考え, その中でも知識の課題に関しては Facebook Logic という新しい論理を提案している. Facebook Logic は従来の知識・信念の論理とは違い, エージェントの集団にエージェント同士がどのような関係性があるかを反映することができる. Facebook Logic の Facebook での友人関係をエージェントのチャネルの有無と読み替えることでエージェント間通信にも応用できる. 通信によるエージェントの変化については Pimolluck ら [6] が commitment の告知と permission の告知を定義しており, これらの告知はエージェントに「~ せよ」や「~ してもよい」といった情報を送ることで, 受信側のエージェントの世界間のアクセス関係を削除や追加を行うことが可能である. commitment の告知については佐野ら [3] によってチャネル通信を扱うことができるように定義されているが, permission の告知はチャネルを用いた定義されておらずチャネルの変化に対応することができていない.

1.2　研究の目的

本研究では, チャネル通信を扱える commitment の告知を元にチャネル通信に対応した permission の告知を定義する. チャネル通信を現実問題で考えたとき, 故障などにより通信チャネルが使えない場合も考えられる. そこで告知による通信チャネルの追加・削除を扱う論理を定義し, エージェント間通信の変化にも対応したシステムを構築, 検証を行う. エージェントの関係性の表現に論理 BUL 用いて, エージェント間通信と通信チャネルの

有無の確認通信を形式化する. エージェント間通信については対象のエージェントのチャネルが存在しない場合に, 他のエージェントに通信対象エージェントとの通信チャネルの有無を問い合わせ, もしチャネルが確認できた場合に告知により通信対象のエージェントと間接的な通信を行い, チャネルが確認できない場合は切断されたエージェントの行動を予測して協調行動させる.

1.3　本論文の構成

本論文では 2 章ではマルチエージェントシステムについてと必要な知識として様相論理について説明を行う. また先行研究をージェント間通信, エージェントの推論パターン, 公開告知論理の 3 つに分けて紹介する. 3 章ではエージェント間の論理についてシングルエージェントでの告知による影響を述べ, 次にそれを前提にしてマルチエージェントでのチャネル通信について説明を行う. 4 章では提案した論理を元に論理システムを実装し検証を行い, 5 章では本論文のまとめと今後の課題について示す.

第2章　関連研究

2.1　マルチエージェントシステム

近年マルチエージェントシステムに関する研究は盛んに行われているが, 高玉 [9] によれば, シングルエージェントシステムではなくなぜマルチエージェントシステムなのかについては次のような利点が挙げられる.

- 問題解決能力
  エージェント単体では解決できないことが複数集まることで解決でき, 1 個体の能力が低くても他のエージェントと協力することで高度な能力が必要な仕事も達成できる

- 適応能力
  対象となる問題の規模の拡大, 対称の変更, 複雑化があった場合でも新たにエージェントを追加したり, エージェント間の相互作用を調整することでうまく対処できる

- ロバスト性
  あるエージェントが故障などで使い物にならなくなったとしても, 他のエージェントが代役をすることでシステム全体の停止を防ぐことが可能

- 並列性
  処理が少ない問題では並列かつ非同期に動作できるので, 全体の処理の高速化や効率を向上させることができる

- モジュール性
  マルチエージェントシステムではエージェントごとにモジュール化されているので, エージェントの再利用や組み合わせをすることで設計コストや時間を低く抑えることが可能

マルチエージェントシステムは分散人工知能の 1 分野として確立しており, 具体的には分散問題解決とマルチエージェントシステムから構成される.

- 分散問題解決
  問題を解決するためにエージェントにどの程度の仕事を割り振るかや分散されたエージェントの結果をいかにして統合して解を求めるかを研究する分野

- マルチエージェントシステム
  問題を解決するために複数のエージェントが各々の知識, 目標, 技術, 計画を駆使して知的な行動を生み出すかを研究する分野

マルチエージェントシステムにおける一般的な研究対象はエージェントを取り巻く分散環境, 他のエージェントとの相互作用などがある. 分散環境は 4 種類に分けることができ, 地理的に異なる場所に存在するものを意味する空間的分散, 時間的にずれたときに発生する時間的分散, 異なった言語体系やオントロジーで利用される意味的分散, 異なった知覚, 行動などの能力で利用される機能的分散がある. 相互作用はレベル, 持続性, 頻度, パターン, 可変性といった種類に分けられており, エージェント間の相互作用の影響や活用法が研究されている.

沼岡ら [5] によると, エージェントのモデル化についてはいくつか代表的な手法が使われていると述べている. マルチエージェントシステムの研究は人工知能の影響を強く受けており, エージェントの計算モデルは, 知識や信念を扱うモデル化技法と密接な関係を持っている. このような技法としてはクリプキによって定式化された可能世界のモデルに基づいた知識と信念の論理やゲーム理論などが挙げられている.

2.2 様相論理

マルチエージェントシステムに必要な知識として様相論理があり, ここではその様相論理の構文論, 意味論, 公理系について述べる.

2.2.1 構文論

東条 [10] によると, 様相論理とは命題変数や命題結合子に加えて様相演算子 $\Box$ や $\Diamond$ が加わった論理であると定義している. $\Box$ はいろいろな呼び方があり, $\Box A$ としたときには「必ず A」や「必然的に A」, $\Diamond$ は $\Diamond A$ としたときに「A かもしれない」や「A である可能性がある」などと読むことができる. 複数の可能世界があり, 全ての世界を同じように観察することができる場合には $\Box$ や $\Diamond$ は「全ての可能世界で」, 「ある可能世界で」という意味に読むことができる. 様相論理の言語と論理式は以下のように定められている. Prop を命題変数の集合とする.

- 言語
  - 命題変数: $\mathsf{Prop} = \{p, q, r, \cdots\}$
  - 命題結合子: $\rightarrow, \vee, \wedge, \neg$
  - 様相記号: $\Box, \Diamond$
- 論理式

$$\varphi ::= \varphi \mid \neg\varphi \mid \varphi \wedge \psi \mid \varphi \vee \psi \mid \varphi \rightarrow \psi \mid \Box\varphi \mid \Diamond\varphi \tag{2.1}$$

2.2.2 意味論

クリプキモデル

可能世界の集合を $W(\neq \emptyset)$, 世界同士にどのようなアクセス可能性 (リンク) があるかを表した二項関係を $R \subseteq W \times W$, 命題変数が真か偽かを定める付値関数を $V(p, w) \in \{0, 1\}$ としたときに, これらをクリプキモデルは対 $\mathfrak{M} = (W, R, V)$ で定義することができる. 例として

- $W = \{w_0, w_1, w_2\}$
- $R = \{(w_0, w_0), (w_0, w_1), (w_0, w_2), (w_1, w_1), (w_2, w_2)\}$
- $\mathfrak{M}, w_0 \not\models p, \mathfrak{M}, w_1 \models p, \mathfrak{M}, w_2 \not\models p,$

とクリプキモデルを定義したとき, モデルをグラフで表すと図 2.1 のように表現できる.

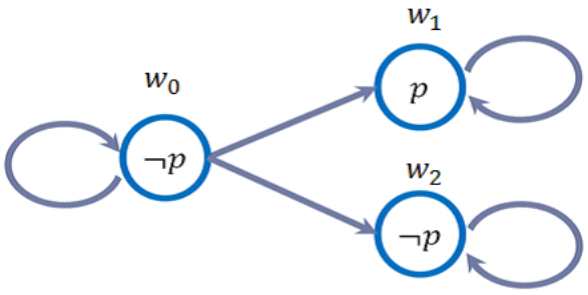


図 2.1 クリプキモデル

モデルでのアクセス可能性の関係をまとめると以下のようになる.

- すべての w で $wRw \Leftrightarrow$ アクセス可能関係が反射的
- すべての w で wRw'ならば $w'Rw \Leftrightarrow$ アクセス可能関係が対称的
- すべての w で wRw'かつ $w'Rw'$ならば $w'Rw'' \Leftrightarrow$ アクセス可能関係が推移的
- すべての w で wRw'かつ wRw''ならば $w'Rw'' \Leftrightarrow$ アクセス可能関係がユークリッド的
- すべての w に w'が存在して $wRw' \Leftrightarrow$ アクセス可能関係が継続的

クリプキ意味論

クリプキモデル $\mathfrak{M} = (W, R, V)$ が与えられたとき, 各世界 $w \in W$ ごとの論理式に以下の真理条件を定める.

(0) $\mathfrak{M}, w \models p \iff V(p, w) = 1$

(1) $\mathfrak{M}, w \models \neg\varphi \iff \mathfrak{M}, w \not\models \varphi$

(2) $\mathfrak{M}, w \models \varphi \wedge \psi \iff \mathfrak{M}, w \models \varphi$ かつ $\mathfrak{M}, w \models \psi$

(3) $\mathfrak{M}, w \models \varphi \vee \psi \iff \mathfrak{M}, w \models \varphi$ あるいは $\mathfrak{M}, w \models \psi$

(4) $\mathfrak{M}, w \models \varphi \to \psi \iff \mathfrak{M}, w \not\models \varphi$ あるいは $\mathfrak{M}, w \models \psi$

(5) $\mathfrak{M}, w \models \Box\varphi \iff$ すべての $w' \in W$ について (wRw'ならば $\mathfrak{M}, w' \models \varphi$)

(6) $\mathfrak{M}, w \models \Diamond\varphi \iff$ ある $w' \in W$ について (wRw'かつ $\mathfrak{M}, w' \models \varphi$)

2.2.3 様相論理式とアクセス可能関係 R の対応

東条 [10] によると, 様相論理には以下のような代表的な論理式がある.

- (T) $\Box\varphi \to \varphi$
- (B) $\varphi \to \Box\Diamond\varphi$
- (4) $\Box\varphi \to \Box\Box\varphi$
- (5) $\Diamond\varphi \to \Box\Diamond\varphi$
- (D) $\Box\varphi \to \Diamond\varphi$

上記の論理式が成り立つということはどういうことかを考える.

T T がモデル $\mathfrak{M}$ のどの可能世界でも成立するとき, 任意の世界 w で $\mathfrak{M}, w \models \Box\varphi$ ならば $\mathfrak{M}, w \models \varphi$ ということになり, 起点となった w でも φ が成り立たなければならない. つまり公理 T が成り立つということは起点となった世界にアクセス可能であることであり, このことを反射的という.

B B が $\mathfrak{M}$ のどの可能世界でも成立するとき, 任意の世界 w で $\mathfrak{M}, w \models \varphi$ ならば $\mathfrak{M}, w \models \Box\Diamond\varphi$ ということになるが, w で φ が成り立つときに w から全てのアクセス可能な世界で $\Diamond\varphi$ が成り立つ必要がある. しかし仮定されているのは $\mathfrak{M}, w \models \varphi$ だけなのでアクセス先の世界で $\Diamond\varphi$ が成り立つには w にアクセス可能である必要がある. つまりアクセス経路があるところにはすべて逆経路が存在することになり, このことを対称的という.

4　4 が $\mathfrak{M}$ のどの可能世界でも成立するとき, 任意の世界 w で $\mathfrak{M}, w \models \Box\varphi$ ならば $\mathfrak{M}, w \models \Box\Box\varphi$ となり, w からアクセス可能なすべての世界で φ が成り立つときにアクセス可能なすべての世界からさらにアクセスできるすべての世界でも φ が成り立つことになる. 成り立たせるためには w からアクセス先の先に直接アクセスできる必要があり, このことを推移的という.

5　5 が $\mathfrak{M}$ で成り立つとき, 任意の世界 w であるアクセス可能な世界で φ が成り立つとして到達関係を wRw', wRw'' とすると w'か w'' のどちらかで φ が成立しなければならない. 仮に w' で φ が成り立つとしたとき, w からアクセス可能なすべての世界で $\Diamond\varphi$ が成り立つので $w'' \models \Diamond\varphi$ であるがこれが成り立つためには $w''Rw'$ が必要である. よって wRw', wRw''ならば $w''Rw'$ でありこのことをユークリッド的という.

D　D が $\mathfrak{M}$ で成り立つときを考える. 任意の世界 w があり w からアクセス可能な世界がない場合に φ を与えると $w \models \Box\varphi, w \not\models \Diamond\varphi$ となる. つまり D が成り立たないときは世界に他の世界へのアクセス関係がないことになる. よって KD が成り立つということは他の世界へアクセス可能であればよい. このことを継続的という.

アクセス可能性を R としてこれらの論理式と R の関係をまとめると以下のようになる.

- R が反射的なら T は $\mathfrak{M}$ のすべての可能世界で成立
- R が対称的なら B は $\mathfrak{M}$ のすべての可能世界で成立
- R が推移的なら 4 は $\mathfrak{M}$ のすべての可能世界で成立
- R がユークリッド的なら 5 は $\mathfrak{M}$ のすべての可能世界で成立
- R が継続的なら D は $\mathfrak{M}$ のすべての可能世界で成立

2.2.4　様相論理のヒルベルト流公理系

様相論理のヒルベルト流公理系を定義する. 様相論理のヒルベルト流公理系 HK は以下の公理と推論規則より成り立つ.

(A1)　$\varphi \to (\psi \to \varphi)$

(A2)　$(\varphi \to (\psi \to \theta)) \to ((\varphi \to \psi) \to (\varphi \to \theta))$

(A3)　$(\neg\varphi \to \neg\psi) \to (\psi \to \varphi)$

(MP)　φ と$\varphi \to \psi$から, ψを導いてもよい

(K)　$\Box(\varphi \to \psi) \to (\Box\varphi \to \Box\psi)$

(Nec)　φ から, $\Box\varphi$ を導いてもよい

次に様相論理のヒルベルト式公理系 HK の証明を定義する.

- 定義
  以下の条件を満たす連続する論理式 $B_1, \ldots, B_n$ を HK の証明とする.
  - B_1は (A1), (A2), (A3), (K) のいずれかであるか,
  - $B_i (1 < i \leqq n)$ は (A1), (A2), (A3), (k) のいずれかであるか, $B_j, B_k (j, k < i)$ から (MP) によって導き出された式であるか, $B_j (j < i)$ から (Nec) によって導き出された式である.

もし論理式 $B_1, \ldots, B_n$ の B_n が B のとき B の証明がある場合, B を HK の定理と呼ぶ. 次に先ほど解説した代表的な論理式とヒルベルト流公理系を用いて新しい公理系を以下のように定義する.

- HKD　(A1), (A2), (A3), (K), (D), (MP), (Nec).
- HKT　(A1), (A2), (A3), (K), (T), (MP), (Nec).
- HKB　(A1), (A2), (A3), (K), (B), (MP), (Nec).
- HS4　(A1), (A2), (A3), (K), (T), (4), (MP), (Nec).
- HS5　(A1), (A2), (A3), (K), (T), (5), (MP), (Nec).

これらの公理系は HK と同じように証明と定理の概念を定義することができる. φ がクリプキモデル $\mathfrak{M}$ で妥当であるとは φ がすべての $w \in W$ について $\mathfrak{M}, w \models \varphi$ となることである. このとき以下の結果 (完全性の健全性) が成立することが知られている.

- $\vdash_{\mathrm{HK}} \varphi \iff$ 全ての $\mathfrak{M}$ においてφ が成り立つ
- $\vdash_{\mathrm{HKD}} \varphi \iff$ 全ての継続的な $\mathfrak{M}$ においてφ が成り立つ
- $\vdash_{\mathrm{HKT}} \varphi \iff$ 全ての反射的な $\mathfrak{M}$ においてφ が成り立つ
- $\vdash_{\mathrm{HKB}} \varphi \iff$ 全ての対称的な $\mathfrak{M}$ においてφ が成り立つ
- $\vdash_{\mathrm{HS4}} \varphi \iff$ 全ての反射的, 推移的な $\mathfrak{M}$ においてφ が成り立つ
- $\vdash_{\mathrm{HS5}} \varphi \iff$ 全ての反射的, 対称的, ユークリッド的な $\mathfrak{M}$ においてφ が成り立つ

2.3 エージェント間通信

2.3.1 FIPA

エージェント間のコミュニケーションの形式化について, エージェント技術の標準化団体 FIPA (Foundation for Intelligent Physical Agents) が形式化した $inform$[2] がある. $inform$ は送信するエージェントについて

- いくつかの命題が真であることを保持している,
- 受信側のエージェントは命題が真であることを信じるように送られてくることを意図している
- すでに受信側エージェントが命題の真理の知識を持っているとは考えない

と定義しており,
$<i$,inform(j,φ)$>$
feasibillity preconditions: $\mathtt{B}_i\varphi \wedge \neg\mathtt{B}_i(\mathtt{Bif}_j\varphi \vee \mathtt{Uif}_j\varphi)$
rational effect: $\mathtt{B}_j\varphi$
と形式化されている. 記号の読み方はそれぞれ,

- $\mathtt{B}_j\varphi$: j はφ と信じる
- $\mathtt{U}_j\varphi$: j はφ についてはっきりとはわからないが$\neg\varphi$ よりはφ だと思っている
- $\mathtt{Bif}_j\varphi$: $\mathtt{B}_j\varphi \vee \mathtt{B}_j\neg\varphi$
- $\mathtt{Uif}_j\varphi$: $\mathtt{U}_j\varphi \vee \mathtt{U}_j\neg\varphi$

となる. つまり i は φ と信じており, かつ i は j は φ について何かしらの信念を持っていることを信じていないときに j は φ と信じるということが言える.

2.3.2 論理体系 BUL

論理体系 BUL は小林ら [4] によって定義された静的なモデルを扱う論理体系 BL の言語 $\mathcal{L}_{BL}$ に動的論理を加えて拡張した論理体系である. この論理によってエージェント間の通信の成否をチャネルの存在と関連付けて形式化することができる. BUL の構文論, 意味論について以下に示す. BUL の言語 $\mathcal{L}_{BUL}$ は, 信念結合子 B と行為演算子 $[inf_{ij}^{\varphi}]$ が用いられており, 論理式 $[inf_{ij}^{\varphi}]\psi$ は inf_{ij}^{φ} が実行された後のモデルで ψ が成り立つという解釈が与えられている. BUL の論理式, 意味論は以下のように定義されている.

論理式

Prop を命題変数の集合, $i, j \in G$ をエージェントの集合とし, BUL の言語 $\mathcal{L}_{BUL}$ は以下のように与える. inf_{ij}^{φ} は i から j へ φ を伝えるという意味となっている.

$$\varphi ::= \top \mid p \mid c_{ij} \mid \neg\varphi \mid \varphi \wedge \psi \mid B_i\varphi \mid [inf_{ij}^{\varphi}\varphi]$$

クリプキ意味論

クリプキモデル $\mathfrak{M}$ は $\mathfrak{M} = \langle W, w_0, \{R_1, \ldots, R_n\}, V\rangle$ で構成されており, W は可能世界の集合, w_0 は起点となる可能世界, R_i は可能世界を繋ぐ信念の到達可能関係, 付値関数 $V(p), V(C_{ij})$ は W の部分集合である. クリプキモデル $\mathfrak{M}$ に対し, $\mathfrak{M}$ の要素と論理式の関係 $\models_{BUL}$ は以下のように定義される. $\mathfrak{M}^{inf_{ij}^{\varphi}}$ は伝達行為 inf_{ij}^{φ} により $\mathfrak{M}$ が更新されたことを示しており, $\mathfrak{M}^{inf_{ij}^{\varphi}} = \langle W, w_0, \{R_1^{inf_{ij}^{\varphi}}, \ldots, R_n^{inf_{ij}^{\varphi}}\}, V\rangle$ である.

$\mathfrak{M}, w \models_{BUL} p \iff w \in V(p)$

$\mathfrak{M}, w \models_{BUL} c_{ij} \iff w \in V(c_{ij})$

$\mathfrak{M}, w \models_{BUL} \neg\varphi \iff \mathfrak{M}, w \not\models_{BUL} \varphi$

$\mathfrak{M}, w \models_{BUL} \varphi \wedge \psi \iff \mathfrak{M}, w \models_{BUL} \varphi$ かつ $\mathfrak{M}, w \models_{BUL} \psi$

$\mathfrak{M}, w \models_{BUL} B_i\varphi \iff \forall w' \; s.t. \; (w, w') \in R_i, \; (\mathfrak{M}, w) \models_{BUL} \varphi$

$\mathfrak{M}, w \models_{BUL} [inf_{ij}^{\varphi}]\psi \iff \mathfrak{M}, w \models_{BUL} c_{ij}$ ならば $\mathfrak{M}^{inf_{ij}^{\varphi}}, w \models_{BUL} \psi$

2.4 エージェントの推論パターン

2.4.1 Facebook Logic

構文論

Seligman ら [8] はコミュニティ内におけるエージェントの知識の変化を捉える論理として Facebook Logic を提案している. Facebook Logic は名前の通りに SNS の Facebook での友人関係のモデル化を行っており, 友人であることや友人関係を表現することができる. Facebook Logic は既存の論理とはエージェントの扱い方に違い, エージェントの集合を与えるだけでなく, エージェント同士が互いにどのような関係性を持っているかが必要となる. マルチエージェントの論理を考える場合にはエージェントごとに様相記号を導入する必要がある. a, b, c をエージェントとすると, $\Box_a, \Box_b, \Box_c$ と表現できる. Facebook Logic は

エージェントに対応する記号がノミナルとして導入されており, 式は世界と個体で真偽が問われる. 既存のマルチエージェント様相論理から Facebook Logic への対応は

$$\Box_a p \text{ が } w \text{ で真} \Longleftrightarrow \Box p \text{ が } (w, a) \text{ で真}$$

とまとめられる. $\Box_a p$ を「a さんが p だと信じる」と読む場合には $\Box p$ は, 「$\sim$ さんは自分が性質 p を持つと信じる」と読み, p はエージェントの性質を表している. Facebook Logic の言語と論理式は以下のように定められている.

- 言語
  - 命題変数: $\mathsf{Prop} = \{p, q, r, \cdots\}$
  - 状態変数: x, y, z, $\cdots$
  - 命題変数: $\neg$, $\wedge$, $\vee$, $\rightarrow$
  - 様相記号: $\Box$, $\Diamond$
  - 関係様相記号 (エージェントの友達関係): $\mathbb{F}$($\sim$ の友達みんなが), $\langle\mathbb{F}\rangle$($\sim$ のある友達が)
  - 充足演算子: $@_a$(a はノミナル), $@_x$(x は状態変数)
  - 局所量化子: $\downarrow$
- 論理式

$$\varphi ::= a|x|p|\neg\varphi|\varphi \wedge \psi|\varphi \vee \psi|\varphi \rightarrow \psi|\Box\varphi||\Diamond\varphi|\mathbb{F}\varphi|\langle\mathbb{F}\rangle\varphi|@_a\varphi| \downarrow x.\varphi.$$

クリプキ意味論

Facebook Logic では世界間のアクセス可能性関係を決める必要がある. 友達関係や誰がどのような性質を持つかは状況によって変わる. クリプキモデル $(W, D, R, \asymp, V)$ は以下のように定義される.

- ありうる状況 (可能世界) の集合 $W(\neq \emptyset)$
- エージェントの集合 $D(\neq \emptyset)$
- エージェントごとの世界間にどのようなアクセス可能性があるかを表した二項関係 $R_d \subseteq W \times W (d \in D)$
- 状況ごとにエージェント間にどのような友達関係があるかを表した二項関係 $\asymp_w \subseteq D \times D (w \in W)$
- 世界とエージェントごとに性質 p が当てはまるかを指定する付値関数 $V(p, (w, d)), V(a, (w, d)) \in \{0, 1\}$

– ただしある $e \in D$ に対して $\{(w,d)|V(a,(w,d))=1\} = W \times \{e\}$, この e は a^V とあらわす.

例として $V(p,(w,d)) = 1$ のときは「エージェント $d \in D$ が性質 p を世界 w でもつ」, $V(a,(w,d)) = 1$ のときは「エージェント $d \in D$ の名前 a である」という意味となる.

Facebook Logic のクリプキモデル $\mathfrak{M} = (W, D, R, \asymp, V)$ が与えられたとき, 各世界とエージェントごとのペア $(w,d) \in W \times D$ の論理式に以下の真理条件が定められる. 論理式 φ がクリプキモデルで妥当であるということは, どんなペア $(w,d) \in W \times D$ についても $\mathfrak{M},(w,d)) \models \varphi$ となる.

(1) $\mathfrak{M},(w,d) \models \neg\varphi \Leftrightarrow \mathfrak{M},(w,d) \not\models \neg\varphi$

(2) $\mathfrak{M},(w,d) \models \varphi \wedge \psi \Leftrightarrow \mathfrak{M},(w,d) \models \varphi$ かつ $\mathfrak{M},(w,d) \models \psi$

(3) $\mathfrak{M},(w,d) \models \varphi \vee \psi \Leftrightarrow \mathfrak{M},(w,d) \models \varphi$ あるいは $\mathfrak{M},(w,d) \models \psi$

(4) $\mathfrak{M},(w,d) \models \varphi \to \psi \Leftrightarrow \mathfrak{M},(w,d) \not\models \varphi$ あるいは $\mathfrak{M},(w,d) \models \psi$

(5) $\mathfrak{M},(w,d) \models \Box\varphi \Leftrightarrow$ すべての $w' \in W$ について (wR_dw'ならば $\mathfrak{M},(w',d) \models \varphi$)

(6) $\mathfrak{M},(w,d) \models \Diamond\varphi \Leftrightarrow$ ある $w' \in W$ について (wR_dw'かつ $\mathfrak{M},(w',d) \models \varphi$)

(7) $\mathfrak{M},(w,d) \models \mathbb{F}\varphi \Leftrightarrow$ すべての $d' \in D$ について ($d \asymp_w d'$ならば $\mathfrak{M},(w',d) \models \varphi$)

(8) $\mathfrak{M},(w,d) \models \langle\mathbb{F}\rangle\varphi \Leftrightarrow$ ある $d' \in D$ について ($d \asymp_w d'$かつ $\mathfrak{M},(w',d) \models \varphi$)

(9) $\mathfrak{M},(w,d) \models @_a\varphi \Leftrightarrow \mathfrak{M},(w,a^V) \models \varphi$

(10) $\mathfrak{M},(w,d) \models{\downarrow}\, x.\varphi \Leftrightarrow \mathfrak{M},(w,d) \models \varphi[\underline{d}/x]$

2.5 公開告知論理

2.5.1 構文論

Plaza [7] はエージェントへの伝達行為によりエージェントの信念の変化を捉える論理について公開告知論理を提案している. 公開告知論理は様相論理に公開告知演算子 $[\varphi!]$ を加えた論理である. $[\varphi!]\psi$ は「φ というアナウンス (告知) があった後に ψ が成り立つ」という意味になる. 公開告知論理の言語と論理式は前節の様相論理の言語と論理に公開告知記号が加わっている.

- 言語
  - 命題変数: $\mathsf{Prop} = \{p, q, r, \cdots\}$

- 命題結合子: $\neg$, $\wedge$, $\vee$, $\rightarrow$
- 様相記号: $\Box_i$, $\Diamond_i$
- 公開告知記号:$[\varphi!]$

- 論理式

$$\varphi ::= p \mid \neg\varphi \mid \varphi \wedge \psi \mid \varphi \vee \psi \mid \varphi \rightarrow \psi \mid \Box_i\varphi \mid \Diamond_i\varphi \mid [\varphi!]\psi$$

2.5.2 意味論

クリプキモデル $\mathfrak{M} = (W, R, V)$ で $[\mathfrak{M}, w \models [\varphi!]\psi]$ をどのように定めるかに関しては, φ が成り立たない世界へのアクセス関係を取り除くことで告知を定義している. 公開告知論理のクリプキ意味論は様相論理の意味論に加えて以下の真理条件が定義される.

$$\mathfrak{M}, w \models [\varphi!]\psi \Leftrightarrow \mathfrak{M}, w \models \varphi \text{ ならば } \mathfrak{M}^{\varphi!}, w \models \psi \tag{2.2}$$

ただし $\mathfrak{M}^{\varphi!}$は対 $(W, R^{\varphi!}, V)$ であり,

$$R^{\varphi!}(x) = R(x) \cap \{w' \in W \mid \mathfrak{M}, w \models \varphi\} \tag{2.3}$$

と定める.

2.5.3 公理系

公開告知論理の公理系について述べる. 公開告知論理の公理系は HK の公理と推論規則に加えて以下の公理系から成り立っている.

(RAtom) $[\varphi!]p \leftrightarrow (\varphi \rightarrow p)$

(R$\neg$) $[\varphi!]\neg\psi \leftrightarrow (\varphi \rightarrow \neg[\varphi!]\psi)$

(R$\rightarrow$) $[\varphi!](\psi \wedge \theta) \leftrightarrow ([\varphi!]\psi \wedge [\varphi!]\theta)$

(R$\Box_i$) $[\varphi!]\Box_i\psi \leftrightarrow (\varphi \rightarrow \Box_i[\varphi!]\psi)$

(R!) $[\varphi!][\psi!]\theta \leftrightarrow [(\varphi \wedge [\varphi!]\psi)!]\theta$

2.5.4　公開告知論理による信念変化

図 2.1 のクリプキモデルに対して $[p!]$ と公開告知があった場合にどのような影響があるかを考える．$[p!]$ によるクリプキモデルの変化は図 2.2 のようになる．点線の矢印は告知により削除されたアクセス関係である．$[p!]$ により全ての世界で $\neg p$ へのアクセス関係が削除されている．図 2.2 の世界 w_0 において p のリンクしかないので信念が $\neg\Box p$ から $\Box p$ に変化している．世界 w_2 でもアクセス関係が無くなったことにより信念が $\neg\Box p\Box p$ に変化していることがわかる．

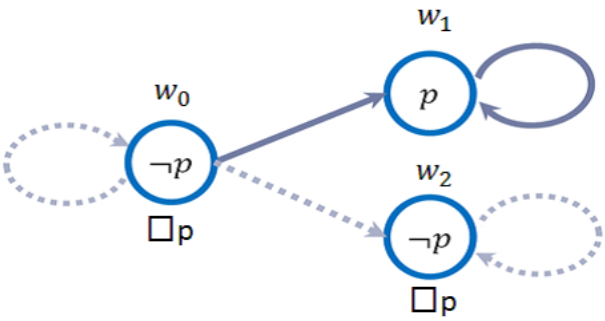


図 2.2 $[p!]$ による信念変化

第3章　エージェント通信のための論理

本章ではエージェント通信のための論理を導入する. まず論理 BUL[4] のアイディアに基づき通信チャネルを導入し, 通信チャネルを用いてマルチエージェントシステムのための論理を構築する. そしてエージェントからエージェントへ通信チャネルを使った告知について解説する. 最後に告知による通信チャネルの追加・削除について定義を行う.

3.1　シングルエージェント

ここではマルチエージェントでのエージェント間通信を定義する前提として伝達行為によるエージェントへの影響をシングルエージェントを想定して説明する.

3.1.1　commitment と permission の告知

Pimolluck ら [6] は commitment の告知として $[\mathtt{com}\ \varphi]$ を, そして新たに permission の告知として $[\mathtt{per}\ \varphi]$ を定義している. commitment と permission の告知について Pimolluck らの論理をシングルエージェントに制限した言語と論理式は以下のようになる.

- 言語
  - 命題変数: $\mathsf{Prop} = \{p, q, r, \cdots\}$
  - 命題結合子: $\neg$, $\wedge$
  - モデル定数: n (この n が現実世界である)
  - 様相記号: $\Box\varphi$ (φ と信じる)
  - 大域記号: $\mathsf{E}\varphi$
  - commitment 告知記号: $[\mathtt{com}\ \varphi]$ (φ せよ)
  - permission 告知記号: $[\mathtt{per}\ \varphi]$ (φ してもよい)
- 論理式

$$\varphi ::= p \mid \mathsf{n} \mid \neg\varphi \mid \varphi \wedge \psi \mid \Box_i\varphi \mid \mathsf{E}\varphi \mid [\mathtt{com}\ \varphi]\psi \mid [\mathtt{per}\ \varphi]\psi \tag{3.1}$$

モデルを $\mathfrak{M}$, 可能世界の集合を W, 世界間のアクセス関係を R_i, 命題変数の真偽を定める付値関数を V, エージェントの集合を G として $\mathfrak{M} = (W, (R_i)_{i \in G}, @, V)$ としたときに様相論理の真理条件に加えて以下のように定義される. commitment の告知では現実の世界のアクセス関係を取り除き, permission の告知は現実の世界のアクセス関係を追加している.

- $\mathfrak{M}, w \models p \Longleftrightarrow w \in V(p)$
- $\mathfrak{M}, w \models \mathsf{n} \Longleftrightarrow w = @$
- $\mathfrak{M}, w \models \neg\varphi \Longleftrightarrow$M,w$\not\models \varphi$
- $\mathfrak{M}, w \models \varphi \wedge \psi \Longleftrightarrow$M,w $\models \varphi$ かつM,w $\models \psi$
- $\mathfrak{M}, w \models \mathsf{E}\varphi \Longleftrightarrow$ ある $u \in W$ について $\mathfrak{M}, u \models \varphi$
- $\mathfrak{M}, w \models [\mathtt{com}\ \varphi]\psi \Longleftrightarrow \mathfrak{M}^{\mathtt{com}\varphi}, w \models \psi$
  - ただし $\mathfrak{M}^{\mathtt{com}\varphi}$は対 $(W, R^{\mathtt{com}\varphi}, S, @, V)$ であり, S はすべての $x \in W$ に対して

$$S(x) = \begin{cases} R(x) \cap [\![\varphi]\!]_{\mathfrak{M}} & (x = @\text{の場合}) \\ R(x) & (\text{それ以外}) \end{cases}$$

    と定める. ここで $[\![\varphi]\!]_{\mathfrak{M}}$ は $[\![\varphi]\!]_{\mathfrak{M}} = \{w \in W | \mathfrak{M}, w \models \varphi\}$ と定め, $\mathfrak{M}$ でφ を真にする全ての世界と定める
- $\mathfrak{M}, w \models [\mathtt{per}^{\downarrow}\ \varphi]\psi \Longleftrightarrow \mathfrak{M}^{\mathtt{per}^{\downarrow}\varphi}, w \models \psi$
  - ただし $\mathfrak{M}^{\mathtt{per}^{\downarrow}\varphi}$は対 $(W, R^{\mathtt{per}^{\downarrow}\varphi}, S', @, V)$ であり, S'はすべての $x \in W$ に対して

$$S'(x) = \begin{cases} R(x) \cup [\![\varphi]\!]_{\mathfrak{M}} & (x = @\text{の場合}) \\ R(x) & (\text{それ以外}) \end{cases}$$

と定める.

3.1.2 commitment の告知による信念変化

図 2.1 のクリプキモデルで $[\mathtt{com}\ p]$ と告知があった場合をどのような信念変化を起こすかを考える. 現実世界は世界 w_0 とする. 告知後のモデルは図 3.1 となった. 世界 w_0 で $[\mathtt{com}\ p]$ が告知された影響で $\neg p$ へのアクセス関係が削除されている. 公開告知論理とは違い, 現実世界のみに影響が出ている為, 世界 w_2 では信念が変化していないことが分かる.

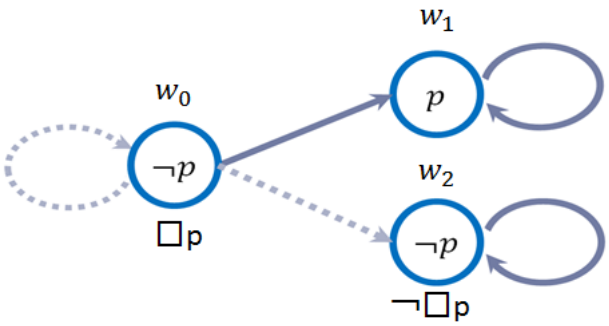


図 3.1 [com p] による信念変化

3.1.3 permission の告知による信念変化

クリプキモデル $\mathfrak{M} = (W, R, V)$ を次のように決める.

- $W = \{w_0, w_1, w_2\}$
- $R = \{(w_0, w_0), (w_0, w_2), (w_2, w_2)\}$
- $\mathfrak{M}, w_0 \not\models p, \mathfrak{M}, w_1 \models p, \mathfrak{M}, w_2 \not\models p$

このモデルをグラフで表すと図 3.2 のように表現できる.

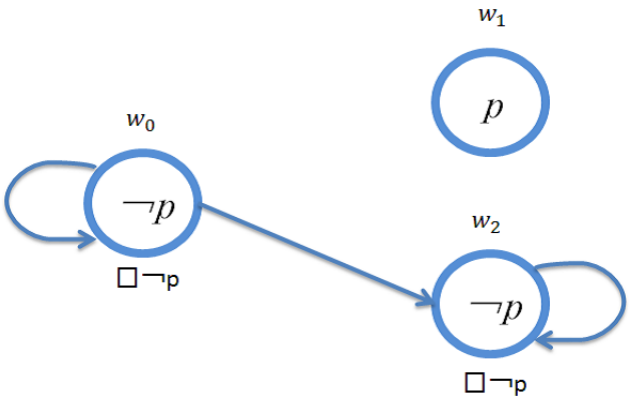


図 3.2 クリプキモデル $\mathfrak{M}$

図 3.2 のクリプキモデルで [per p] と告知があった場合をどのような信念変化を起こすかを考える. 現実世界は世界 w_0 とする. 告知後のモデルは図 3.1 となった. 赤色の二重線矢印は告知により追加されたアクセス関係を表している. 世界 w_0 で [per p] が告知された

影響で p へのアクセス関係が追加されている. commitment の告知と同じく現実世界のみに影響が出ている為, 世界 w_1 では信念が変化していないことが分かる.

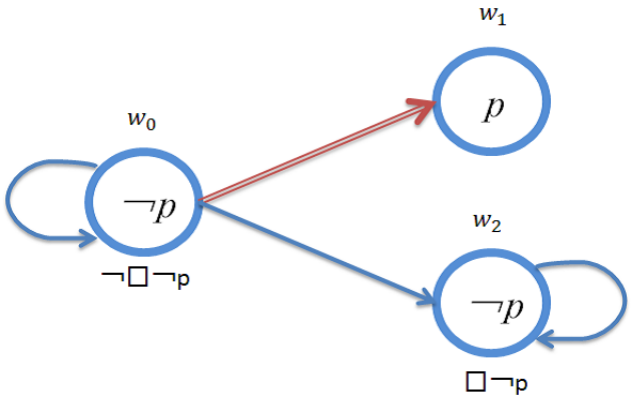


図 3.3 [per p] による信念変化

3.2 マルチエージェント

本節では前節で解説した告知についてチャネル通信を用いてマルチエージェントに対応せるとともに, 告知によるチャネルの追加・削除について定義を行う.

3.2.1 チャネル通信

告知はエージェント間の通信であるのでシングルエージェントでの告知は必ず成功するとしてきたが, エージェント間を通信チャネル [4] で繋いでマルチエージェント化させた場合, 告知は通信チャネル経由で送られるべきであり. チャネルの有無により告知の成否が変わることが考えられる. Pimolluck ら [6] が定義した [com φ] と [per φ] はマルチエージェントを想定して考えられているが, チャネル通信には対応しておらず告知は必ず成功するようになっている. 佐野ら [3] は [com φ] に対してチャネル通信を扱っており, それをもとに Pimolluck らの定義をチャネル通信を扱えるように新しく定義を行う. まずエージェントからエージェントへ告知を送信して信念を変えることを考える. エージェントをそれぞれ i と j で定義したときに告知を送信のためにはエージェント i からエージェント j に通信チャネルが存在する必要がある. エージェント i からエージェント j へ ⌈φ である⌋ と告知が現実になされた後でψ は $[\mathtt{com}^{\downarrow^i_j}\varphi]\psi$, エージェント i からエージェント j へ ⌈φ である可能性がある⌋ と告知が現実になされた後でψ は $[\mathtt{per}^{\downarrow^i_j}\varphi]\psi$ と表現できる.

今までは告知により世界間のアクセス関係を追加・削除を行っていたが, チャネル通信も導入に伴って告知によってチャネルの追加や削除もできるべきだと考えられる. チャネルを追加・削除することによって今まで告知を送ることができなかったエージェントにも

告知を送ることができ, 他のエージェントへの告知を遮断することが可能となる. 告知によるチャネルの追加・削除は他の告知と同じように現実でなされると考え, 告知を送るためにはチャネルが必要となる. チャネルを追加するときは, チャネルを繋ぎたいエージェントと他のエージェントを介して間接的にチャネルが繋がっている場合のみであり, 介するエージェントは 1 つまでとする. ある現実世界 w_0 にエージェント i, j, k が存在し, ⌈エージェント i はエージェント k を介してエージェント j へチャネルを追加する⌋と告知がなされた場合には, 下記の図 3.4 のようにエージェント k を介して間接的にチャネルが繋がっている必要がある.

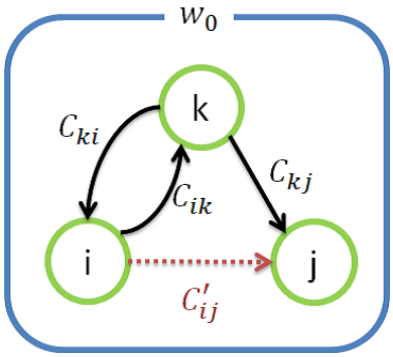


図 3.4 告知によるチャネルの追加

チャネルを追加する場合にはエージェント k が⌈エージェント i にエージェント j へのチャネルを追加する⌋と告知を行うとする.言語と論理式について以下のように定義する. エージェントの集合を $G = \{i, j, k\}$ とする.
言語

- 命題論理の言語
- モデル定数: n (現実世界は n である)
- チャネル: c_{ij} (エージェント i はエージェント j と通信可能である)
- 様相記号: $\Box_i\varphi$ (i がφ と信じる)
- commitment 告知: $[\mathrm{com}^{\downarrow \mathrm{i}}_{\mathrm{j}}\varphi]\psi$ (i から j に⌈φ である⌋という告知が現実でなされた後にψ)
- permission 告知: $[\mathrm{per}^{\downarrow \mathrm{i}}_{\mathrm{j}}\varphi]\psi$ (i から j に⌈φ である可能性もある⌋という告知が現実でなされた後にψ
- チャネル追加告知: $[\mathtt{C}^{+}_{k}(j,i)]\psi$ (⌈j は k を介して i と間接的にチャネルが繋がっている場合, i へのチャネルを追加する⌋という告知が現実でなされた後にψ)
- チャネル削除告知: $[\mathtt{C}^{-}(j,i)]\psi$ (⌈j は i へのチャネルを削除する⌋という告知が現実でなされた後にψ)

- 論理式

$$\varphi ::= p \mid \neg\varphi \mid c_{ij} \mid \varphi \wedge \psi \mid \varphi \vee \psi \mid \varphi \to \psi \mid \mathsf{n} \mid \Box_i\varphi \mid \Diamond_i\varphi \mid \mathsf{E}\varphi \mid$$
$$[\mathrm{com}^{\downarrow \mathrm{i}}_{\mathrm{j}}\ \varphi]\psi \mid [\mathrm{per}^{\downarrow \mathrm{i}}_{\mathrm{j}}\ \varphi]\psi \mid [\mathtt{C}^{+}_{k}(j,i)]\psi \mid [\mathtt{C}^{-}(j,i)]\psi$$

モデルを $\mathfrak{M}$, 可能世界の集合を W, 世界間のアクセス関係を R_i, 命題変数の真偽を定める付値関数を $V, V(p), V(c_{ij})$ を W の部分集合, エージェントの集合を G として
$\mathfrak{M} = (W, (R_i)_{i\in G}, (C_{ij})_{i,j\in G}, @, V)$ としたときに C_{ij} と $[\mathtt{com}^{\downarrow^i_j}\varphi]\psi$ と $[\mathtt{per}^{\downarrow^i_j}\varphi]\psi$ は以下のように真偽条件を定義する.

- $\mathfrak{M}, w \models \mathsf{C}_{ij}$

$$\mathfrak{M}, w \models \mathsf{C}_{ij} \Longleftrightarrow w \in C_{ij} \tag{3.2}$$

- $\mathfrak{M}, w \models [\mathtt{com}^{\downarrow^i_j}\varphi]\psi$

$$\mathfrak{M}, w \models [\mathtt{com}^{\downarrow^i_j}\varphi]\psi \Longleftrightarrow \mathfrak{M}^{\mathtt{com}^{\downarrow^i_j\varphi}}, w \models \psi \tag{3.3}$$

ただし $\mathfrak{M}^{\mathtt{com}^{\downarrow^i_j\varphi}}$ は対 $(W, (R'_k)_{k\in G}, (C_{ij})_{i,j\in G}, @, V)$ であり, $(R'_k)_{k\in G}$ は以下のように定める.

  - $k \neq j$ のとき,

$$R'_k := R_k \tag{3.4}$$

  - $k = j$ のとき, $x \in W$ に対して,

$$R'_j(x) = \begin{cases} R_j(x) \cap [\![\varphi]\!]_{\mathfrak{M}} & (x = @\text{かつ } x \in C_{ij}\text{の場合}) \\ R_j(x) & (\text{それ以外}) \end{cases} \tag{3.5}$$

$[\![\varphi]\!]_{\mathfrak{M}} = \{w \in W | \mathfrak{M}, w \models \varphi\}$ であり $\mathfrak{M}$ で φ を真にする全状況と定める

- $\mathfrak{M}, w \models [\mathtt{per}^{\downarrow^i_j}\varphi]\psi$

$$\mathfrak{M}, w \models [\mathtt{per}^{\downarrow^i_j}\varphi]\psi \Longleftrightarrow \mathfrak{M}^{\mathtt{per}^{\downarrow^i_j\varphi}}, w \models \psi \tag{3.6}$$

ただし $\mathfrak{M}^{\mathtt{per}^{\downarrow^i_j\varphi}}$ は対 $(W, (R'_k)_{k\in G}, (C_{ij})_{i,j\in G}, @, V)$ であり, $(R'_k)_{k\in G}$ は以下のように定める.

– $k \neq j$ のとき,

$$R'_k := R_k \tag{3.7}$$

– $k = j$ のとき, $x \in W$ に対して,

$$R'_j(x) = \begin{cases} R_j(x) \cup [\![\varphi]\!]_{\mathfrak{M}} & (x = @\text{かつ } x \in C_{ij}\text{の場合}) \\ R_j(x) & (\text{それ以外}) \end{cases} \tag{3.8}$$

$[\![\varphi]\!]_{\mathfrak{M}} = \{w \in W | \mathfrak{M}, w \models \varphi\}$ であり $\mathfrak{M}$ でφ を真にする全状況と定める

告知によるチャネルの追加・削除の真理条件は以下のように与える.

- $\mathfrak{M}, w \models [\mathtt{C}_k\mathtt{+}(i,j)]\psi$

$$\mathfrak{M}, w \models [\mathtt{C}^{+}_{k}(i,j)]\psi \Longleftrightarrow \mathfrak{M}^{\mathtt{C}^{+}_{k}(i,j)}, w \models \psi \tag{3.9}$$

ただし $\mathfrak{M}^{\mathtt{C}^{+}_{k}(i,j)} = (W, (R_l)_{l\in G}, (C'_{mn})_{m,n\in G}, @, V)$ であり C'_{mn} は,

– C_{ik} と C_{kj} が現実で成立するとき,

$$C'_{ij} := \{@\} \cup C_{ij} \tag{3.10}$$

– C_{ik} か C_{kj} のどれか 1 つでも現実で成立しないとき,

$$C'_{ij} := C_{ij} \tag{3.11}$$

- $\mathfrak{M}, w \models [\mathtt{C}\mathtt{-}(i,j)]\psi$

$$\mathfrak{M}, w \models [\mathtt{C}^{-}(i,j)]\psi \Longleftrightarrow \mathfrak{M}^{\mathtt{C}^{-}(i,j)}, w \models \psi \tag{3.12}$$

ただし $\mathfrak{M}^{\mathtt{C}^{-}(i,j)} = (W, (R_l)_{l\in G}, (C'_{mn})_{m,n\in G}, @, V)$ であり

$$C'_{ij} = C_{ij} - \{@\} \tag{3.13}$$

第4章　実装

本章では 3 章で提案した論理に基づいてチャネルによるエージェント間通信に対応した推論システムについて述べる．このシステムはクリプキモデルのデータと論理式を入力し, 論理式の真偽とクリプキモデルの画像を出力する．システムは Java, JavaCC, Prolog, Graphviz で実装し, 入力された論理式の解析に JavaCC, 論理計算に Prolog, モデル画像の生成に Graphviz, 他のプログラミング言語への連携に Java を使用している．システムの具体的な処理の流れは以下の通りである．

1. 論理式解析
2. 論理式計算
3. クリプキモデル更新
4. モデル画像出力

4.1　システムの構成

本システムは Java を中心に構成されており, 解析や論理計算などそれぞれ得意な処理がある他のプログラミング言語と連携を取っている．例としてモデル $\mathfrak{M}$ として $\lceil \mathfrak{M}, w0 \models [\mathtt{com}^{\downarrow_j^i} p](\Box_j p) \rfloor$ をシステムで検証する．モデルのデータは Java のソースコードに記入し, 論理式はキーボードで入力する．システムへの入力は Java で構築した入力画面にて「 [com (i-=>j_p)](□ <j_p>),w0」 と入力する．入力を行うと入力された式を解析する前にモデルのデータを Prolog のシステムに送り, 初期状態のモデル画像とチャネル関係図の生成に必要な dot ファイルを作成する．このモデルの dot ファイルは可能世界, アクセス関係, 付値関数から構成され, チャネル関係図の dot ファイルにはモデルの dot ファイルを構成する 3 要素に加えてチャネル関係から構成されている．次に入力された式は JavaCC で構築したシステムに送られ式の解析を行う．ここでは式を告知, 論理式, 対象となる世界に分類し, 論理式が括弧で囲われている場合には変数に置き換える．式の分析, 変換が終わると式は Java のシステムに返され, 次に式の計算を行うため Prolog で構築したシステムに送られる．Prolog のシステムでは式の計算及びモデルの画像を生成するために必要な dot ファイルを生成する．式の計算と dot ファイルの生成を行うと式の計算結果を Java のシステム

に返す．その後 Java のシステムは計算結果を出力するともに dot ファイルより画像を生成する Graphviz システムが行うべき処理をまとめた dat ファイルを生成し, dat ファイルより Graphviz を動作させる．Graphviz は dot ファイルを元に初期状態のモデルの画像とチャネル関係図, モデルが更新されている場合はその更新モデルの画像とチャネル関係図を作成する．システムの構成及び処理の流れを以下に示す．丸で囲った数字は Java のシステムが処理を行う順番である．

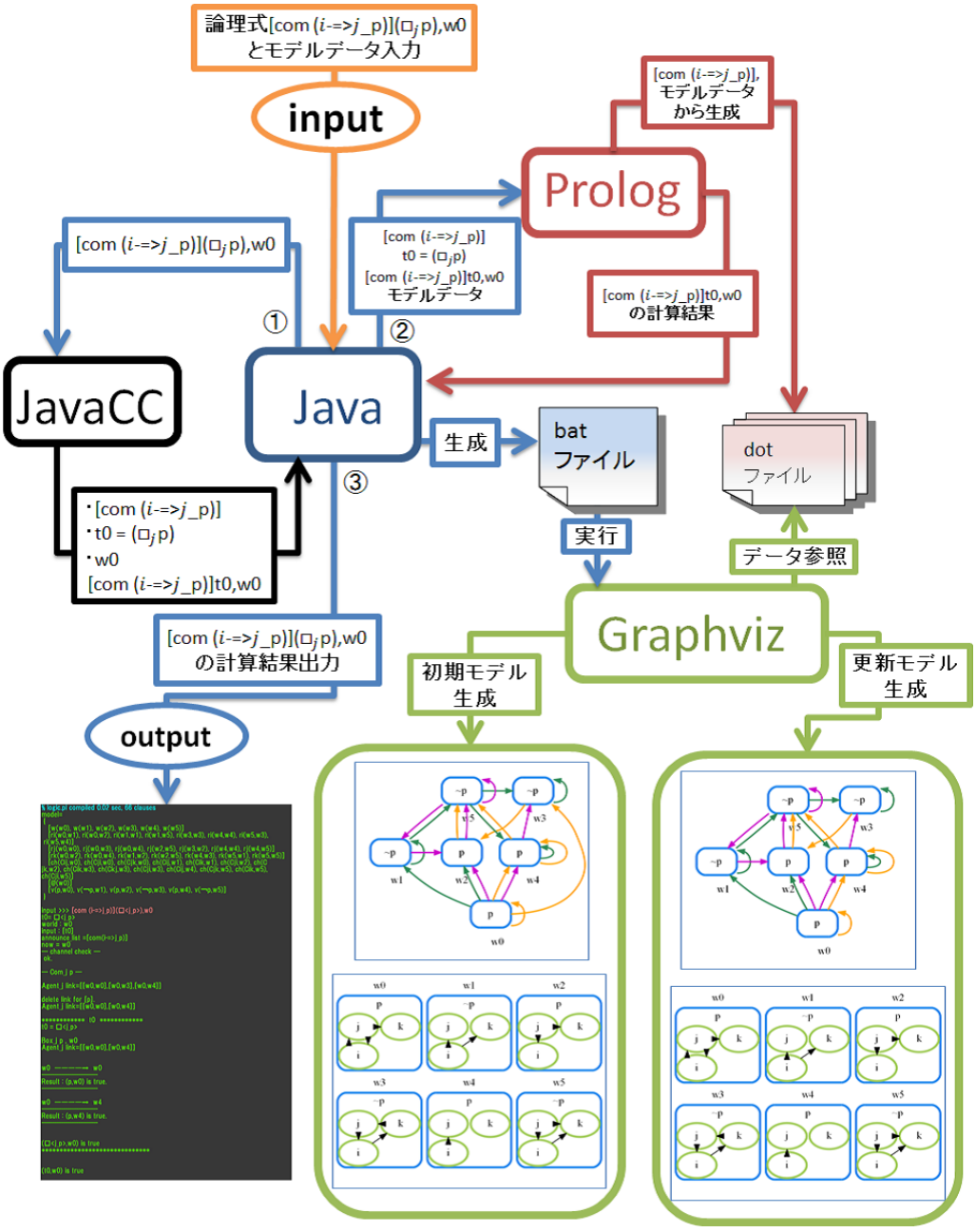


図 4.1 システムの処理の流れ

4.2 モデル

本システムはシングルエージェントとマルチエージェントに対応しており, 入力するクリプキモデルは以下のように定義する. シングルエージェントで告知を行う場合にはチャネルがないので告知は必ず送信されるとする.

- シングルエージェント

$$\mathfrak{M}_{\text{single}} = (W, R, @, V) \tag{4.1}$$

- マルチエージェント

$$\mathfrak{M}_{\text{multi}} = (W, (R_i)_{i\in G}, (C_{ij})_{i,j\in G}, @, V) \tag{4.2}$$

図 4.2 に実際に入力するモデルのデータの例を示す. モデルデータはシステム起動時に自動的に読み込まれる. w は世界, r は到達可能関係, v は各世界の付値関数, ch はエージェント同士のチャネル関係, @は現実の世界を表しており ˜ は ¬ の意味とする.

```
model = [
    { w(w0), w(w1), w(w2), w(w3) },
    { r(w0,w0,j,j_color), r(w0,w1,j,j_color), r(w1,w3,j,j_color) },
    { r(w0,w2,d,d_color), r(w2,w2,d,d_color),r(w2,w1,d,d_color),
      r(w2,w3,d,d_color),r(w3,w3,d,d_color) },
    { v(p,w0), v(˜p,w1), v(p,w2), v(˜p,w3) },
    { ch(cjd,w0,j,d), ch(cdj,w0,d,j), ch(cdj,w1,d,j),
      ch(cjd,w2,j,d), ch(cjd,w3,j,d), ch(cdj,w3,d,j) },
    { @(w0) } ];
```

図 4.2 モデルの入力例

モデルで使われている関数の読み方と構成する要素の見方について示す. マルチエージェントの場合には各エージェントのアクセス可能関係を色によって区別しているので r にはモデル生成用に必要な色の要素が, チャネル ch にもモデル生成用に必要なチャネル元のエージェントとチャネル先のエージェントの要素が含まれている.

- $\mathtt{w}(w) \Longrightarrow$「可能世界 w が存在する」
  - w ··· 可能世界
- $\mathtt{r}(w_0, w_1, i, i\_color) \Longrightarrow$「エージェント i は世界 w_0から世界 w_1までアクセス可能である」

- w_0 ··· アクセス可能関係のアクセス元の可能世界の要素
- w_1 ··· アクセス可能関係のアクセス先の可能世界の要素
- i ··· エージェント
- i_color ··· モデル画像生成用のエージェントを識別するための色要素

- $\mathtt{v}(v, w) \Longrightarrow$「世界 w ではφ である」
  - v ··· 付値関数
  - w ··· 付値関数がどの可能世界の付値関数かを示す要素

- $\mathtt{ch}(C_{ij}, w, i, j) \Longrightarrow$「世界 w でエージェント i からエージェント j へ通信可能である」
  - C_{ij} ··· チャネル
  - w ··· チャネルがどの可能世界に存在するかを示す要素
  - i ··· チャネル元のエージェントの識別要素
  - j ··· チャネル先のエージェントの識別要素

- $\mathtt{@}(w) \Longrightarrow$「可能世界の中で現実の世界は w である」
  - w ··· 現実の世界

次に生成されるモデル画像例について解説する. モデル画像例として図 4.2 のデータを入力して生成された画像を図 4.3 に示す. 青色の四角形は世界を, 四角形の中にはその世界の付値関数を, 矢印はアクセス可能関係を示している. アクセス可能関係はエージェントごとに色分けされて画像出力される.

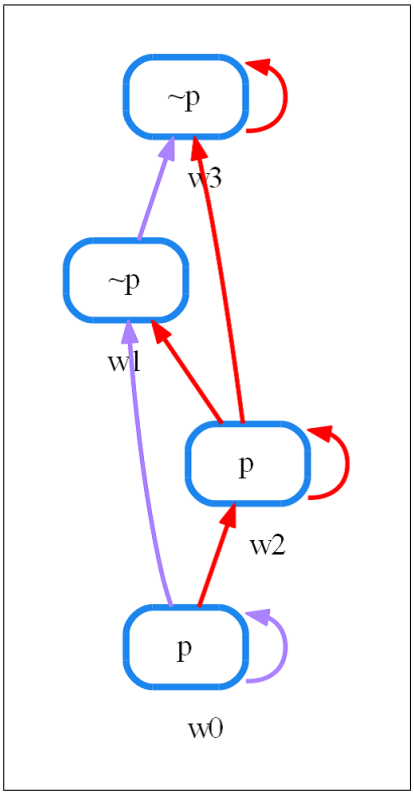


図 4.3 モデル画像の出力例

マルチエージェントの場合には上記のモデル画像に加えてチャネルの繋がりを表したチャネル関係図も生成される．生成される画像例として図 4.4 を示す．モデルの画像と同様に青色の四角形は世界を表しており, 緑色の円はエージェントで黒色の矢印はチャネルを表している．

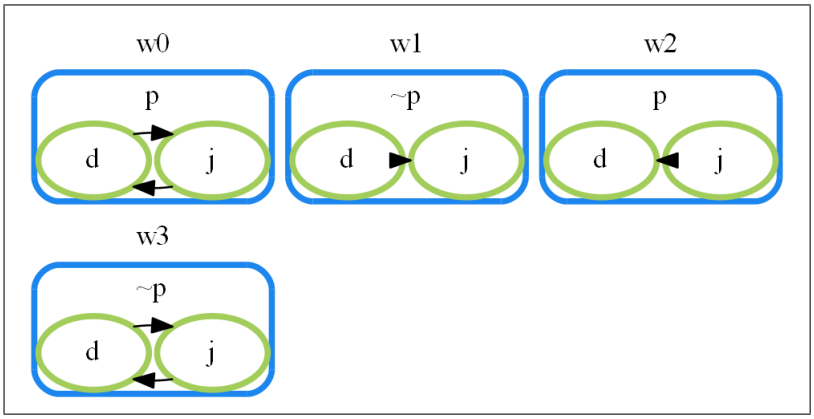


図 4.4 チャネル関係図の出力例

これらの画像は論理式を入力したときに初期状態のモデル画像が生成され, 告知が行われてモデルの更新があった場合には更新モデルの画像も生成される. システムは告知が行われるたびに画像が生成する. 例としてモデルを $\mathfrak{M}$ として $\mathfrak{M}, w \models [\mathtt{com}\ \varphi][\mathtt{com}\ \psi][\mathtt{com}\ \theta]\delta$を入力した場合, 初期状態のモデル $\mathfrak{M}, \mathfrak{M}$ が $[\mathtt{com}\ \varphi]$ の影響を受けて更新したモデル $\mathfrak{M}^{\mathtt{com}\ \varphi}, \mathfrak{M}^{\mathtt{com}\ \varphi}$ が $[\mathtt{com}\ \psi]$ の影響を受けて更新したモデル $\left(\mathfrak{M}^{\mathtt{com}\ \varphi}\right)^{\mathtt{com}\ \psi}, \left(\mathfrak{M}^{\mathtt{com}\ \varphi}\right)^{\mathtt{com}\ \psi}$が $[\mathtt{com}\ \theta]$ の影響を受けて更新したモデル $\left(\left(\mathfrak{M}^{\mathtt{com}\ \varphi}\right)^{\mathtt{com}\ \psi}\right)^{\mathtt{com}\ \theta}$, 計 4 つのモデルの画像が生成される.

4.3 システムで扱える論理式

システムで扱うことができる論理式について以下に定義する. i, j, k をエージェントすると,

- $\Box_i\varphi$: i がφ と信じる
- $\Diamond_i\varphi$: i が$\neg\varphi$ と信じる
- $[\mathtt{com}^{\downarrow_j^i}\varphi]\psi$: i から j に「φ である」という告知が現実でなされた後にψ
- $[\mathtt{per}^{\downarrow_j^i}\varphi]\psi$: i から j に「φ である可能性もある」という告知が現実でなされた後にψ
- $[\mathtt{C}_k^+(j,i)]\psi$:
  「j は k を介して i と間接的にチャネルが繋がっている場合, i へのチャネルを追加する」という告知が現実でなされた後にψ
- $[\mathtt{C}^-(j,i)]\psi$:
  「j は i へのチャネルを削除する」という告知が現実でなされた後にψ

システムで扱える論理式は以下のとおりである.

- シングルエージェント

$$\varphi ::= p \mid \neg\varphi \mid \varphi \wedge \psi \mid \varphi \vee \psi \mid \varphi \to \psi \mid \mathsf{n} \mid \Box\varphi \mid \Diamond\varphi \mid [\mathtt{com}^{\downarrow}\ \varphi]\psi \mid [\mathtt{per}^{\downarrow}\ \varphi]\psi$$

- マルチエージェント

$$\varphi\ ::= p \mid \neg\varphi \mid \mathsf{c}_{ij} \mid \varphi \wedge \psi \mid \varphi \vee \psi \mid \varphi \to \psi \mid \mathsf{n} \mid \Box_i\varphi \mid \Diamond_i\varphi \mid [\mathtt{com}^{\downarrow_j^i}\ \varphi]\psi \mid [\mathtt{per}^{\downarrow_j^i}\ \varphi]\psi \mid [\mathtt{C}_k^+(j,i)]\psi \mid [\mathtt{C}^-(j,i)]\psi$$

実際にシステムで入力する場合には上記で定義した演算子を次のように変換し入力する.

- $\Box_i\varphi \implies$ □<i_φ>

- $\Diamond_i\varphi \implies$ `<i_φ>`
- $[\mathtt{com}^{\downarrow^i_j}\varphi]\psi \implies$ `[com (i->j_φ)]`ψ
- $[\mathtt{per}^{\downarrow^i_j}\varphi]\psi \implies$ `[per (i->j_φ)]`ψ
- $[\mathtt{C}^+_k(i,j)]\psi \implies$ `[add_ch (i-k-j)]`ψ
- $[\mathtt{C}^-(i,j)]\psi \implies$ `[del_ch (Cij)]`ψ

例としてあるモデル $\mathfrak{M}$ に対して $\mathfrak{M}, w0 \models [\mathrm{com}^{\downarrow^i_j}p][\mathrm{per}^{\downarrow^i_j}\neg p](\Box_j p)$ という論理式を入力する場合, `[com (i->j_p)][per (i->j_¬p)](□<j_p>),w0` と入力する.

4.4 アルゴリズム

システムで扱うことができる演算子のアルゴリズムを以下に示す. 第 1 引数を s, 第 2 引数を t と定義した時, findall は findall(s,t) と表現している. 処理は t が成立する場合の s をリストに追加していき最後にそのリストを出力する. forall は forall(s,t) と表現し,forall の第 1 引数 s がとりうる全ての解が第 2 引数 t で成立するときに true を返す. 1 つでも t が成立しない場合は false を返す.

Algorithm 1 : $\Box_j\psi$ の処理

input $\mathfrak{M} = (W, R_j, R_d, @, V), \Box_j\psi, w$
findall $(w' \in W,\ w_0 R_j w')$
  add w' **to** X
end findall
forall $(w' \in X,\ (\mathfrak{M}, w' \models \psi))$
  $\mathfrak{M}, w_0 \models \Box_j\psi$
else
  $\mathfrak{M}, w_0 \not\models \Box_j\psi$
end forall

Algorithm 1 は □ の処理である. まず findall では $w' \in W, w_0 R_j w'$ を引数とし, w_0 から到達可能である世界の集合であるリスト X を作成する. 作成した X を使い, forall で w_0 から到達できる世界を w' とし, 全ての w' で $\mathfrak{M}, w_0 \models \psi$ が成立するかの処理を行う.

Algorithm 2 : $[\mathtt{com}^{\downarrow_i^j}(\psi)]$ の処理

input $\mathfrak{M}_1 = (W, (R_i)_{i\in G}, @, V)$**,** $[\mathbf{com}_i(\psi)], w$
if$(@ \in C_{ji})$
  findall $(w' \in W,\ @R_i w')$
    add w' **to** X
  end findall
  findall $(w' \in X,\ (\mathfrak{M}, w' \models \varphi))$
    add w' **to** Y
  end findall
end if
$R_i' := (R_i \setminus \{(@, w') | w' \in X\}) \cup \{(@, w') | w' \in Y\}$
output $\mathfrak{M}_2 = (W(R_i')_{i\in G}, @, V)$

Algorithm 2 は告知によるリンク削除処理である. 最初に現実の世界において告知を送るエージェントへのチャネルが存在するかを確認する. もしチャネルが存在しない場合には告知は送られないとする. チャネルが存在する場合には, まず現実の世界から到達可能である世界をリスト X にまとめる. 次に $w' \in W$ かつ $\mathfrak{M}, w' \models \psi$ が成立する w' を調べ, Y に追加する. 次に告知により削除されたリンクの集合 R_j' を作成しモデルを更新する.

Algorithm 3 : $[\mathtt{per}^{\downarrow_i^j}(\varphi)]$ の処理

input $\mathfrak{M}_1 = (W, (R_i)_{i\in G}, @, V)$, $[\mathbf{per}_i(\varphi)]$**,** w
if$(@ \in C_{ji})$
  findall $(w' \in W,\ ((\mathfrak{M}_1, w' \models \varphi) \wedge \neg(@R_i w')))$
    add w' **to** Y
  end findall
end if
$R_i' := R_i \cup \{(@, w') | w' \in Y\}$
output $\mathfrak{M}_2 = (W, (R_i')_{i\in G}, @, V)$

Algorithm 3 は告知によるリンク追加処理である. Algorithm 2 と同じように, 告知を送るエージェントへのチャネルが存在するかを確認する. チャネルが存在する場合には, φ が成り立ちかつ現実世界とまだリンクが繋がっていない世界をリスト Y にまとめる. 次に Y を利用して告知によって追加されたリンクを含めた集合 R_j' を作成しモデルを更新する.

Algorithm 4：$[\mathtt{C^-}(i,j)]$ の処理

input $\mathfrak{M}_1 = (W, (R_i)_{i\in G}, (C_{ij})_{i,j\in G}, @, V)$, $[\mathtt{C^+}(i,j)^{\downarrow_i^j}], w$
if$(@ \in C_{ij})$
  $C' := C_{ij} \setminus \{@\}$
else
  $C'_{ij} := C_{ij}$
end if
output $\mathfrak{M}_2 = (W, (R_i)_{i\in G}, (C'_{ij})_{i,j\in G}, @, V)$

Algorithm 4 は告知によるチャネルの削除処理である．削除したいチャネルの確認を行い，そのチャネルがある場合にはチャネル削除を行いモデルを更新する．告知を送るエージェントへのチャネルや削除したいチャネルがない場合にはチャネルの変更は行われないとする．

Algorithm 5：$[\mathtt{C}^+_k(i,j)]$ の処理

input $\mathfrak{M}_1 = (W, (R_i)_{i\in G}, (C_{ij})_{i,j\in G}, @, V)$, $[\mathtt{C}^+_k(i,j)^{\downarrow_i^j}], w$
if$(@ \in C_{ik} \wedge @ \in C_{kj})$
  if$(@ \in C_{ij})$
    $C'_{ij} = C_{ij}$
  else
    $C'_{ij} = C_{ij} \cup \{@\}$
  end if
end if
output $\mathfrak{M}_2 = (W, (R_i)_{i\in G}, (C'_{ij})_{i,j\in G}, @, V)$

Algorithm 5 は告知によるチャネルの追加処理である．現実の世界において他のエージェントを介してチャネルがチャネルを繋ぎたいエージェントと間接的に繋がっているかを確認する．もし間接的な繋がりが確認できた場合にはチャネルを追加する．ただし告知を送るエージェントへのチャネルや追加したいチャネルがすでに存在している場合にはチャネルの変更は行わないとする．

4.5　シングルエージェントでの実行例

4.5.1　様相演算子

様相演算子を使った論理式の実行例を示す．この実行例ではシングルエージェントのモデルで処理を行った．シングルエージェントで処理を行う場合には様相演算子はそれぞ

れ $\Box_s\varphi, \Diamond_s\varphi$ とし，システムでは $\Box$<s_φ>, $\Diamond$<s_φ> と入力する．クリプキモデル $\mathfrak{M}_1 = (W, R, V)$ を以下のように定義する．

- $W = \{w_0, w_1, w_2, w_3, w_4\}$
- $R = \{(w_0, w_1), (w_0, w_2), (w_0, w_3), (w_0, w_4), (w_2, w_2), (w_4, w_4)\}$
- $@ = w_0$
- $\mathfrak{M}, w_0 \models p, \mathfrak{M}, w_1 \models p, \mathfrak{M}, w_2 \models p, \mathfrak{M}, w_3 \not\models p, \mathfrak{M}, w_4 \not\models p,$

システムの出力結果について解説する．次の画像は出力結果を切り取ったものである．

```
input >>> □<s_p>,w0
world : w0
input : [□<s_p>]

Box_s p , w0
Agent_s link=[[w0,w1],[w0,w2],[w0,w3],[w0,w4]]
```

図 4.5 入力・解析処理

赤字の部分が入力する論理式である．論理式を入力しエンターキーを押すとシステムが論理式を解析し，論理式と対象とする世界を識別する．論理式に様相演算子が含まれている場合には図 4.5 の下部のように世界間のアクセス関係も表示される．次の図 4.6 は計算処理結果の一部である．

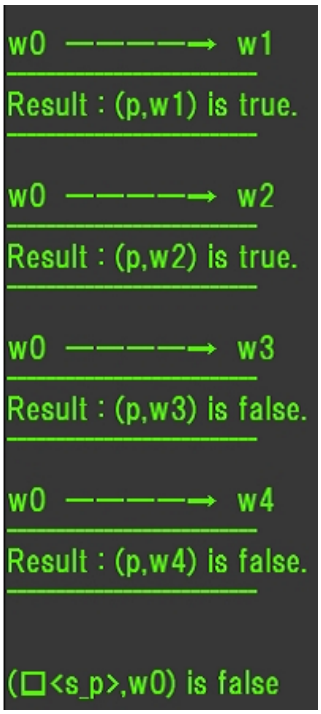


図 4.6 計算処理

図 4.6 に表示されている「w0 ---→ w1」は w0 から w1 にアクセスしていることを表現しており, アクセスした後に $\mathfrak{M}_1, w1 \models p$ の処理を行っている．様相演算子 ◇ はシステムでは ¬□¬ として扱っており論理式の解析処理で変換を行っている．次に実行例として $\mathfrak{M}_1, w_0 \models \Box_s p$ と $\mathfrak{M}_1, w_0 \models \Diamond_s p$ を入力した図 4.7 と図 4.8 に示す.

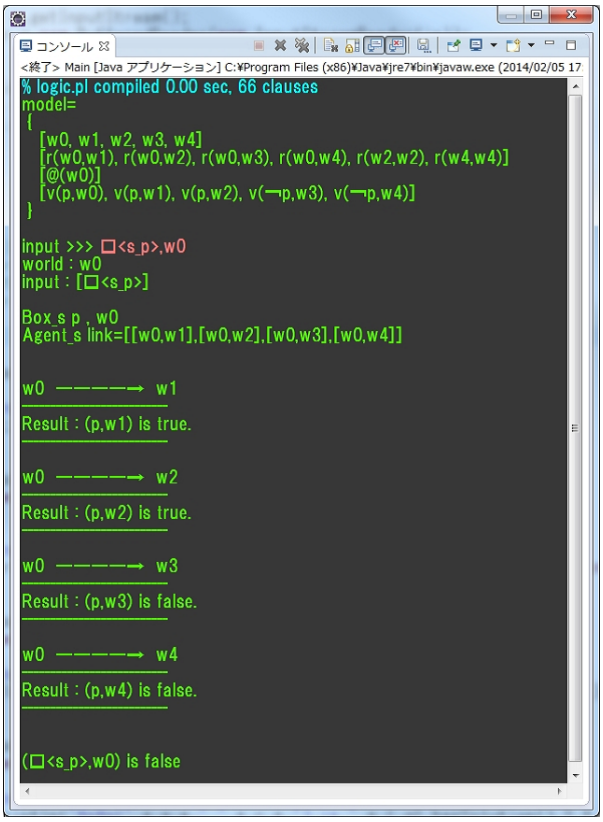


図 4.7 $\mathfrak{M}_1, w_0 \models \Box_s p$ の出力結果

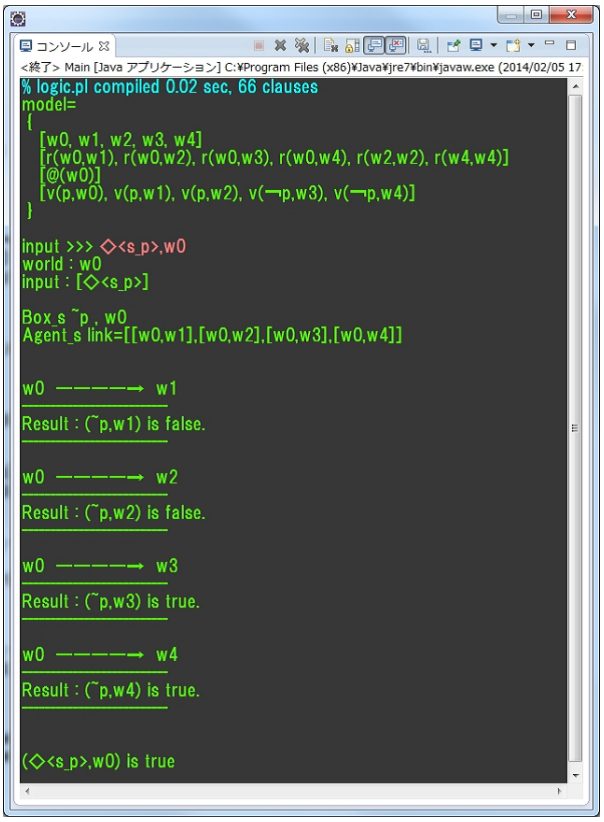


図 4.8 $\mathfrak{M}_1, w_0 \models \Diamond_s p$ の出力結果

出力結果について, $\mathfrak{M}_1, w_0 \models \Box_s p$ と $\mathfrak{M}_1, w_0 \models \Diamond_s p$ はそれぞれ $false$ と $true$ になるが, 結果を見るとそれぞれ正しい結果を出力していることが分かる. 演算子の処理についてそれぞれの論理式が指定した世界から到達可能な世界にアクセスして処理を行っていることが確認できる. システムにより生成された $\mathfrak{M}_1$ の画像を図 4.9 に示す.

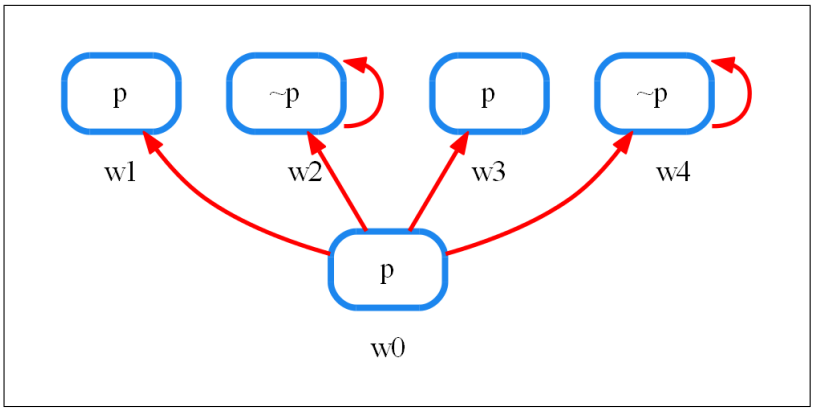


図 4.9 生成された $\mathfrak{M}_1$ の画像

4.5.2 告知

告知演算子を使った論理式の実行例を示す. 先ほど定義したクリプキモデル $\mathfrak{M}$ に対して [com] と [per] の告知を行いモデルを更新させる. シングルエージェントの場合には告知は $[\mathtt{com}^{\downarrow_s^s}\varphi]$ とし告知は必ず成功するとする. 次に告知演算子を使った場合の出力結果についてを解説する. 図 4.10 は告知演算子を使った場合の出力結果の一部である. 最初にチャネルの有無を確認し, もしチャネルが存在する場合には告知の処理を行う. 次に現実世界でのリンクが表示され, その後告知によりリンクの更新を行う.

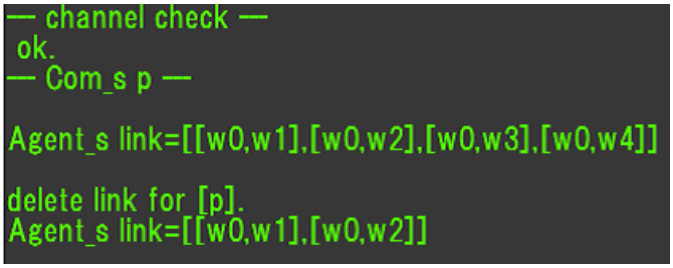


図 4.10 告知の処理

次に告知の実行例として告知による世界間のアクセス関係の削除や追加を行う. システムで $\mathfrak{M}_1, w_0 \models [\mathtt{com}^{\downarrow_s^s}p](\Box_s p)$ と $\mathfrak{M}_1, w_0 \models [per^{\downarrow_s^s}p](\Box_s p)$ を入力し, その結果と更新されたモデル画像を次に示す. 本システムでは丸括弧で囲った式は変数で置き換えて計算を行っている. 今回の実行例では $(\Box_s p)$ を t0 に置き換えて計算を行っている.

```
% logic.pl compiled 0.02 sec, 66 clauses
model=
 {
   [w0, w1, w2, w3, w4]
   [r(w0,w1), r(w0,w2), r(w0,w3), r(w0,w4), r(w2,w2), r(w4,w4)]
   [@(w0)]
   [v(p,w0), v(p,w1), v(p,w2), v(¬p,w3), v(¬p,w4)]
 }

input >>> [com (s-=>s_p)](□<s_p>),w0
t0= □<s_p>
world : w0
input : [t0]
announce_list =[com(s-=>s_p)]
now = w0
— channel check —
 ok.
— Com_s p —

Agent_s link=[[w0,w1],[w0,w2],[w0,w3],[w0,w4]]

delete link for [p].
Agent_s link=[[w0,w1],[w0,w2]]

************ t0 ************
t0 = □<s_p>

Box_s p , w0
Agent_s link=[[w0,w1],[w0,w2]]


w0 ————→ w1
————————————
Result : (p,w1) is true.
————————————

w0 ————→ w2
————————————
Result : (p,w2) is true.
————————————


(□<s_p>,w0) is true
*****************************


(t0,w0) is true
```

図 4.11 $\mathfrak{M}_1, w_0 \models [\mathrm{com}^{\downarrow_s^s} p]\Box_s p$ の入力結果

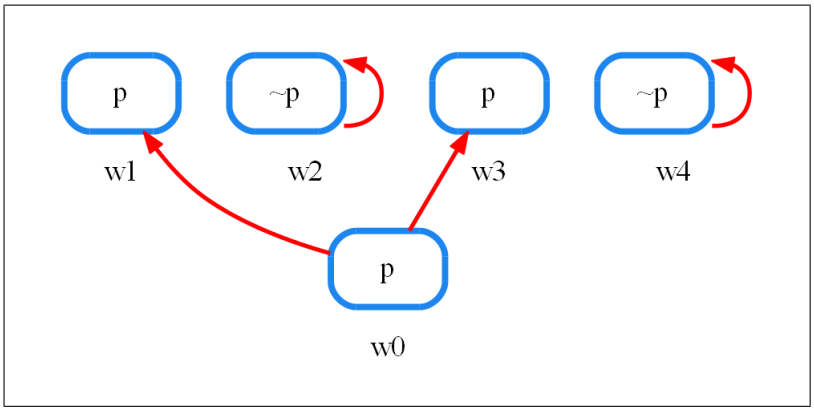


図 4.12 告知により更新されたモデル $\mathfrak{M}_1^{\mathrm{com}^{\downarrow_s^s} p}$

```
% logic.pl compiled 0.02 sec, 66 clauses
model=
 {
   [w0, w1, w2, w3, w4]
   [r(w0,w1), r(w0,w2), r(w0,w3), r(w0,w4), r(w2,w2), r(w4,w4)]
   [@(w0)]
   [v(p,w0), v(p,w1), v(p,w2), v(¬p,w3), v(¬p,w4)]
 }

input >>> [per (s-=>s_p)](□<s_p>),w0
t0= □<s_p>
world : w0
input : [t0]
announce_list =[per(s-=>s_p)]
now = w0
— channel check —
 ok.
— Per_s_p —
Agent_s link=[[w0,w1],[w0,w2],[w0,w3],[w0,w4]]

insert link for [p].
Agent_s link=[[w0,w1],[w0,w2],[w0,w3],[w0,w4],[w0,w0]]

************ t0 ************
t0 = □<s_p>

Box_s_p , w0
Agent_s link=[[w0,w1],[w0,w2],[w0,w3],[w0,w4],[w0,w0]]


w0 ————→ w1
————————
Result : (p,w1) is true.
————————

w0 ————→ w2
————————
Result : (p,w2) is true.
————————

w0 ————→ w3
————————
Result : (p,w3) is false.
————————

w0 ————→ w4
————————
Result : (p,w4) is false.
————————

w0 ————→ w0
————————
Result : (p,w0) is true.
————————


(□<s_p>,w0) is false
*******************************

(t0,w0) is false
```

図 4.13 $\mathfrak{M}_1, w_0 \models [\mathrm{per}^{\downarrow_s^s} p]\Box_s p$ の入力結果

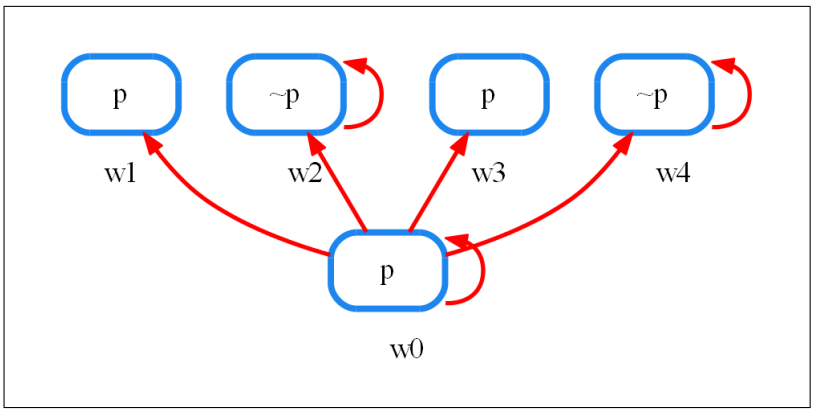


図 4.14 告知により更新されたモデル $\mathfrak{M}_1^{\mathrm{per}^{\downarrow_s^s p}}$

結果よりそれぞれ告知によりリンクが削除・追加され, 現実の世界において $\Box_s p$ のアクセスする世界が変わっていることが分かる. $w_0 \models [\mathrm{com}^{\downarrow_s^s} p]\Box_s p$ では図 4.7 の出力結果と比べると告知の影響を受け結果が変わっており, モデル画像も告知の影響を受けてモデルが更新されていることが確認できる.

4.6 マルチエージェントでの実行例

4.6.1 チャネルの告知

告知によるチャネルの更新の実行例を示す. ここからはマルチエージェントのモデルで処理を行う. G をエージェントの集合としてクリプキモデル $\mathfrak{M}_2 = (W, (R_i)_{i\in G}, (C_{ij})_{i,j\in G}, @, V)$ を定義する.

- $W = \{w_0, w_1, w_2, w_3, w_4, w_5\}$
- $R_i = \{(w_0, w_1), (w_0, w_2), (w_1, w_1), (w_1, w_5), (w_3, w_3), (w_4, w_4), (w_5, w_3), (w_5, w_4)\}$
- $R_j = \{(w_0, w_0), (w_0, w_3), (w_0, w_4), (w_2, w_5), (w_3, w_2), (w_4, w_4), (w_5, w_5)\}$
- $R_k = \{(w_0, w_2), (w_0, w_4), (w_1, w_2), (w_2, w_5), (w_4, w_1), (w_5, w_1), (w_5, w_1)\}$
- $C_{ij} = \{w_0, w_1, w_4\}$
- $C_{ik} = \{w_1, w_3, w_5\}$
- $C_{ji} = \{w_0, w_2, w_3, w_5\}$
- $C_{jk} = \{w_0, w_2, w_5\}$
- $C_{kj} = \{w_3\}$
- $@ = \{w_0\}$
- $\mathfrak{M}, w_0 \models p, \mathfrak{M}, w_1 \not\models p, \mathfrak{M}, w_2 \models p, \mathfrak{M}, w_3 \not\models p, \mathfrak{M}, w_4 \models p, \mathfrak{M}, w_5 \not\models p$

告知によるチャネルの更新の出力画像についてに示す. 図 4.15 は告知によるチャネル削除の出力画像の一部である. 図 4.15 の上部に告知前のチャネルの集合が表示されている. 中央の ⌈ `--- del channel cjk ---` ⌋ はチャネルの削除処理を表しており, この例では C_{jk} を削除している. 図 4.15 の下部に告知後のチャネルの集合が表示されている.

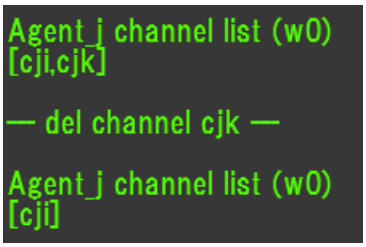


図 4.15 告知によるチャネル削除処理

図 4.16 は告知によるチャネル追加の出力画像を切り取ったものである．チャネルを追加する場合には繋ぎたいエージェントと間接的にチャネルが繋がっている必要があるので「 `--- connect channel check ---` 」では間接的なチャネルが存在するかの確認を行っている．もし間接的なチャネルが確認できた場合には図の下部の「 `--- add channel ---` 」からの処理が行われチャネルが追加される．

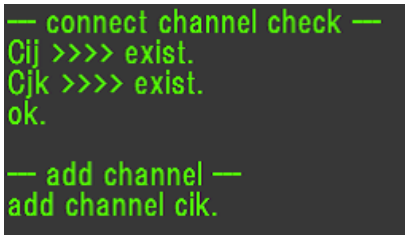


図 4.16 告知によるチャネルの追加処理

次にモデル $\mathfrak{M}_2$ をシステムに入力したときに生成される告知前のモデル画像とチャネル関係図を以下に示す．リンクについては緑色の矢印はエージェント i, オレンジ色の矢印はエージェント j, 紫色の矢印はエージェント k とする．

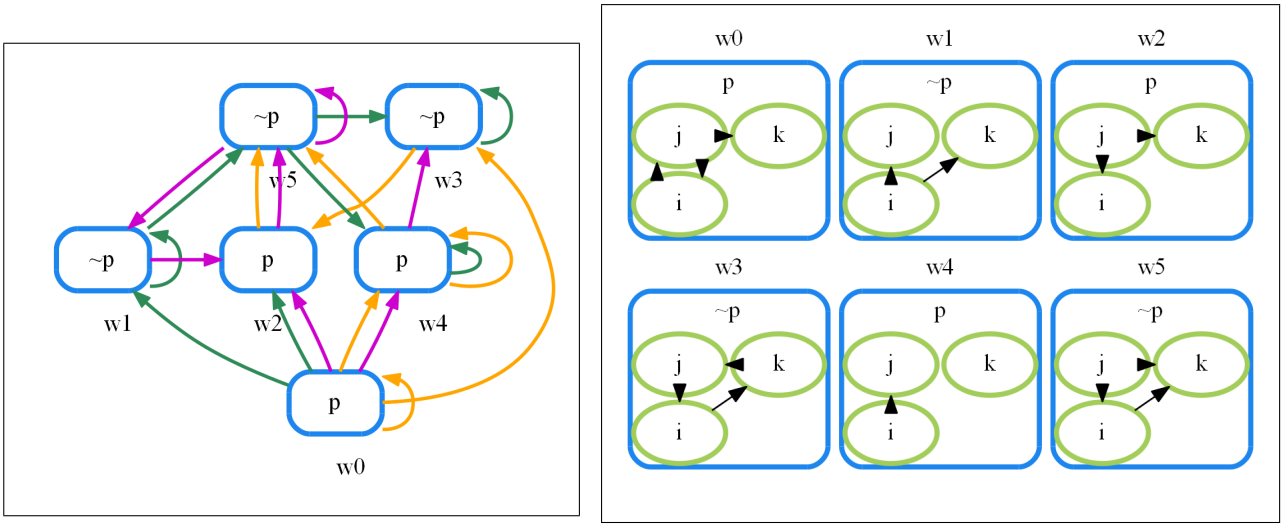


図 4.17 生成されたモデル $\mathfrak{M}_2$ の画像とチャネル関係図

今回, 実行例として $\mathfrak{M}_2, w_0 \models [\mathsf{C}^-(j,k)]\mathsf{C}_{jk}$ と $\mathfrak{M}_2, w_0 \models [\mathsf{C}^+_j(i,k)]\mathsf{C}_{ik}$ の入力を行った. モデルが告知による影響を比較するために $\mathfrak{M}_2, w_0 \models \mathsf{C}_{jk}$ と $\mathfrak{M}_2, w_0 \models \mathsf{C}_{ik}$ の入力結果を図 4.18 と図 4.19 に示す.

```
% logic.pl compiled 0.00 sec, 66 clauses
model=
{
   [w(w0), w(w1), w(w2), w(w3), w(w4), w(w5)]
   [ri(w0,w1), ri(w0,w2), ri(w1,w1), ri(w1,w5), ri(w3,w3), ri(w4,w4), ri(w5,w3),
ri(w5,w4)]
   [rj(w0,w0), rj(w0,w3), rj(w0,w4), rj(w2,w5), rj(w3,w2), rj(w4,w4), rj(w4,w5)]
   [rk(w0,w2), rk(w0,w4), rk(w1,w2), rk(w2,w5), rk(w4,w3), rk(w5,w1), rk(w5,w5)]
   [ch(Cij,w0), ch(Cji,w0), ch(Cjk,w0), ch(Cij,w1), ch(Cik,w1), ch(Cji,w2), ch(C
jk,w2), ch(Cik,w3), ch(Ckj,w3), ch(Cji,w3), ch(Cij,w4), ch(Cjk,w5), ch(Cik,w5),
ch(Cji,w5)]
   [@(w0)]
   [v(p,w0), v(¬p,w1), v(p,w2), v(¬p,w3), v(p,w4), v(¬p,w5)]
}

input >>> Cjk,w0
world : w0
input : [Cjk]


(Cjk,w0) is true
```

図 4.18 $\mathfrak{M}_2, w_0 \models \mathsf{C}_{jk}$ の入力結果

```
% logic.pl compiled 0.02 sec, 66 clauses
model=
{
   [w(w0), w(w1), w(w2), w(w3), w(w4), w(w5)]
   [ri(w0,w1), ri(w0,w2), ri(w1,w1), ri(w1,w5), ri(w3,w3), ri(w4,w4), ri(w5,w3),
ri(w5,w4)]
   [rj(w0,w0), rj(w0,w3), rj(w0,w4), rj(w2,w5), rj(w3,w2), rj(w4,w4), rj(w4,w5)]
   [rk(w0,w2), rk(w0,w4), rk(w1,w2), rk(w2,w5), rk(w4,w3), rk(w5,w1), rk(w5,w5)]
   [ch(Cij,w0), ch(Cji,w0), ch(Cjk,w0), ch(Cij,w1), ch(Cik,w1), ch(Cji,w2), ch(C
jk,w2), ch(Cik,w3), ch(Ckj,w3), ch(Cji,w3), ch(Cij,w4), ch(Cjk,w5), ch(Cik,w5),
ch(Cji,w5)]
   [@(w0)]
   [v(p,w0), v(¬p,w1), v(p,w2), v(¬p,w3), v(p,w4), v(¬p,w5)]
}

input >>> Cik,w0
world : w0
input : [Cik]


(Cik,w0) is false
```

図 4.19 $\mathfrak{M}_2, w_0 \models \mathsf{C}_{ik}$ の入力結果

$\mathfrak{M}_2, w_0 \models [\mathsf{C}^-(j,k)]\mathsf{C}_{jk}$ と $\mathfrak{M}_2, w_0 \models [\mathsf{C}^+_j(i,k)]\mathsf{C}_{ik}$ の入力結果を図 4.20 と図 4.21 に示す.

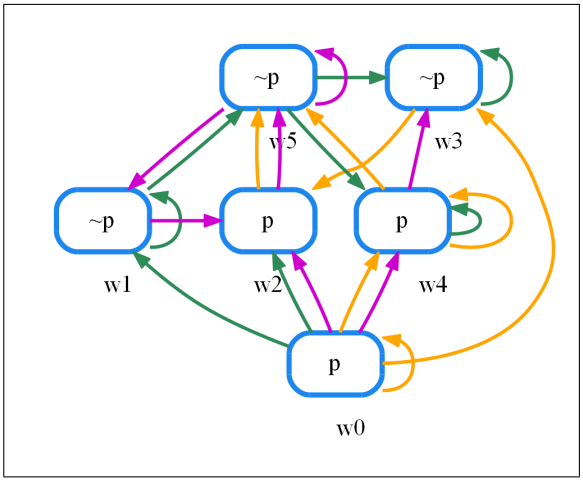
```
% logic.pl compiled 0.02 sec, 66 clauses
model=
{
  [w(w0), w(w1), w(w2), w(w3), w(w4), w(w5)]
  [ri(w0,w1), ri(w0,w2), ri(w1,w1), ri(w1,w5), ri(w3,w3), ri(w4,w4), ri(w5,w3),
ri(w5,w4)]
  [rj(w0,w0), rj(w0,w3), rj(w0,w4), rj(w2,w5), rj(w3,w2), rj(w4,w4), rj(w4,w5)]
  [rk(w0,w2), rk(w0,w4), rk(w1,w2), rk(w2,w5), rk(w4,w3), rk(w5,w1), rk(w5,w5)]
  [ch(Cij,w0), ch(Cji,w0), ch(Cjk,w0), ch(Cij,w1), ch(Cik,w1), ch(Cji,w2), ch(C
jk,w2), ch(Cik,w3), ch(Ckj,w3), ch(Cji,w3), ch(Cij,w4), ch(Cjk,w5), ch(Cik,w5),
ch(Cji,w5)]
  [@(w0)]
  [v(p,w0), v(¬p,w1), v(p,w2), v(¬p,w3), v(p,w4), v(¬p,w5)]
}

input >>> [del_ch (Cjk)](Cjk),w0
t0= Cjk
world : w0
input : [t0]
announce_list =[del_ch(Cjk)]
now = w0

 channel list (w0)
[clj,cji,cjk]

— del channel cjk —

 channel list (w0)
[clj,cji]

************ t0 ************
t0 = Cjk


(Cjk,w0) is false
******************************


(t0,w0) is false
```

図 4.20 $\mathfrak{M}_2, w_0 \models [\mathsf{C}^-(j,k)]\mathsf{C}_{jk}$の入力結果

```
% logic.pl compiled 0.02 sec, 66 clauses
model=
{
  [w(w0), w(w1), w(w2), w(w3), w(w4), w(w5)]
  [ri(w0,w1), ri(w0,w2), ri(w1,w1), ri(w1,w5), ri(w3,w3), ri(w4,w4), ri(w5,w3),
ri(w5,w4)]
  [rj(w0,w0), rj(w0,w3), rj(w0,w4), rj(w2,w5), rj(w3,w2), rj(w4,w4), rj(w4,w5)]
  [rk(w0,w2), rk(w0,w4), rk(w1,w2), rk(w2,w5), rk(w4,w3), rk(w5,w1), rk(w5,w5)]
  [ch(Cij,w0), ch(Cji,w0), ch(Cjk,w0), ch(Cij,w1), ch(Cik,w1), ch(Cji,w2), ch(C
jk,w2), ch(Cik,w3), ch(Ckj,w3), ch(Cji,w3), ch(Cij,w4), ch(Cjk,w5), ch(Cik,w5),
ch(Cji,w5)]
  [@(w0)]
  [v(p,w0), v(¬p,w1), v(p,w2), v(¬p,w3), v(p,w4), v(¬p,w5)]
}

input >>> [add_ch (i-j-k)](Cik),w0
t0= Cik
world : w0
input : [t0]
announce_list =[add_ch(i-j-k)]
now = w0
— connect channel check —
Cij >>>> exist.
Cjk >>>> exist.
ok.

— add channel —
add channel cik.

************ t0 ************
t0 = Cik


(Cik,w0) is true
******************************


(t0,w0) is true
```

図 4.21 $\mathfrak{M}_2, w_0 \models [\mathsf{C}_j^+(i,k)]\mathsf{C}_{ik}$の入力結果

結果より $\mathfrak{M}_2, w_0 \models \mathsf{C}_{jk}$ と $\mathfrak{M}_2, w_0 \models \mathsf{C}_{ik}$ はそれぞれ true と false を出力しているが, $\mathfrak{M}_2, w_0 \models [\mathsf{C}^-(j,k)]\mathsf{C}_{jk}$ と $\mathfrak{M}_2, w_0 \models [\mathsf{C}_j^+(i,k)]\mathsf{C}_{ik}$ はそれぞれ false と true を出力しており, 告知によりチャネルがそれぞれ削除, 追加されているのが確認できる. 次に生成されたモデル画像とチャネル関係図を以下に示す.



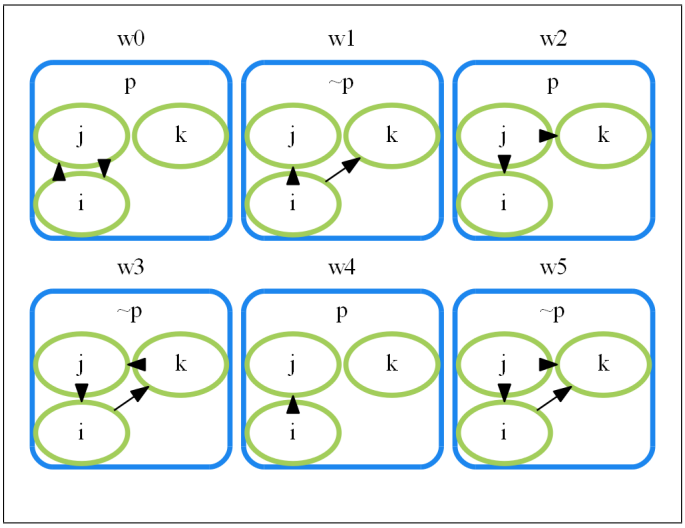


図 4.22 生成されたモデル $\mathfrak{M}_2^{\mathrm{del\_ch}^{C_{jk}}}$ の画像とチャネル関係図

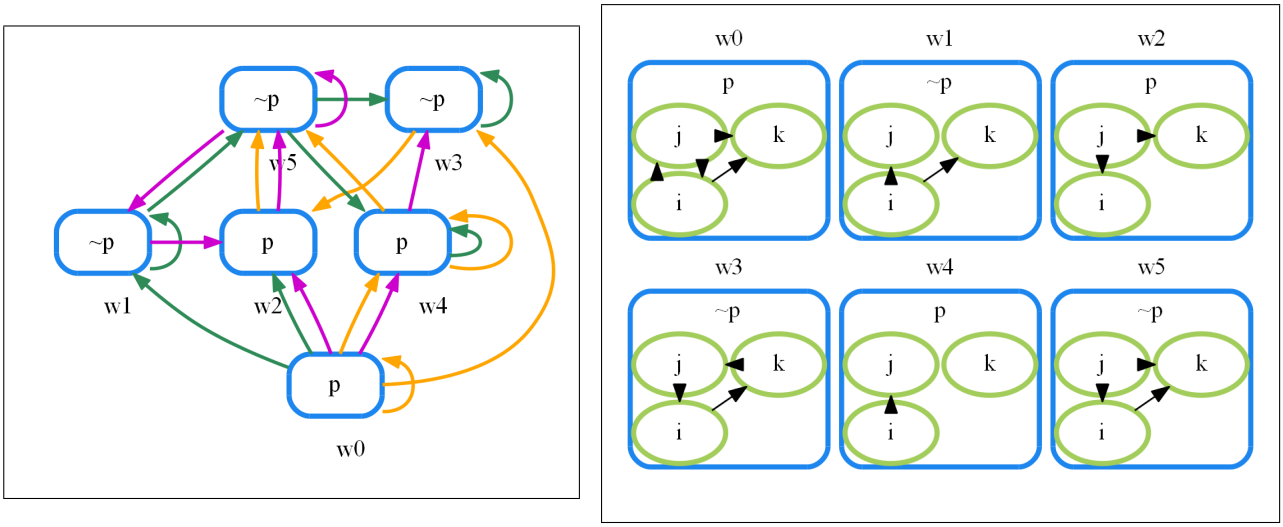


図 4.23 生成されたモデル $\mathfrak{M}_2^{\mathrm{add\_ch}_j^{C_{ik}}}$ の画像とチャネル関係図

図 4.22 では初期状態のチャネル相関図と比べて $w0$ で C_{jk} が削除されており, 図 4.23 では $w0$ で C_{ik} が追加されていることが分かる. 今回は告知によるチャネルの変更なのでアクセス関係には影響はなく, 図 4.22 と図 4.23 のモデル画像には初期状態のモデル画像と比べても変化がないのがわかる.

4.6.2 すべての演算子の組み合わせ

今まで定義した演算子である様相演算子, commitment 告知演算子, permission 告知演算子, チャネル追加告知演算子, チャネル削除告知演算子, これらをすべてを使った実行例を示す. システムにて $\mathfrak{M}_2, w_0 \models [\mathtt{com}^{\downarrow_i^j} C_{ij}][\mathsf{C}_j^+(i,k)][\mathtt{per}^{\downarrow_k^i}\neg p][\mathsf{C}^-(i,j)]((\Box_i\Diamond_i p) \vee (\neg \mathsf{C}_{ij}))$ を入力した. 入力結果を図 4.24 と図 4.25, 生成されたグラフ画像とチャネル関係図を図 4.26 にに示す. これよりクリプキモデル $\mathfrak{M}_3, \mathfrak{M}_4, \mathfrak{M}_5, \mathfrak{M}_6$ を更新モデルとして定義し, 以下のように意味を与える.

- $\mathfrak{M}_3 = \mathfrak{M}_2^{\mathrm{com}^{\downarrow_i^j}\mathsf{C}_{ij}}$
- $\mathfrak{M}_4 = \left(\mathfrak{M}_2^{\mathrm{com}^{\downarrow_i^j}\mathsf{C}_{ij}}\right)^{\mathsf{C}_j^+(i,k)}$
- $\mathfrak{M}_5 = \left(\left(\mathfrak{M}_2^{\mathrm{com}^{\downarrow_i^j}\mathsf{C}_{ij}}\right)^{\mathsf{C}_j^+(i,k)}\right)^{\mathtt{per}^{\downarrow_k^i}\neg p}$
- $\mathfrak{M}_6 = \left(\left(\left(\mathfrak{M}_2^{\mathrm{com}^{\downarrow_i^j}\mathsf{C}_{ij}}\right)^{\mathsf{C}_j^+(i,k)}\right)^{\mathtt{per}^{\downarrow_k^i}\neg p}\right)^{\mathsf{C}^-(j,i)}$

```
% logic.pl compiled 0.02 sec, 66 clauses
model=
 [
   [w(w0), w(w1), w(w2), w(w3), w(w4), w(w5)]
   [ri(w0,w1), ri(w0,w2), ri(w1,w1), ri(w1,w5), ri(w3,w3), ri(w4,w4), ri(w5,w3), ri(w5,w4)]
   [rj(w0,w0), rj(w0,w3), rj(w0,w4), rj(w2,w5), rj(w3,w2), rj(w4,w4), rj(w4,w5)]
   [rk(w0,w2), rk(w0,w4), rk(w1,w2), rk(w2,w5), rk(w4,w3), rk(w5,w1), rk(w5,w5)]
   [ch(Cij,w0), ch(Cji,w0), ch(Cjk,w0), ch(Cik,w1), ch(Cji,w2), ch(Cjk,w2), ch(Cik,w3),
ch(Ckj,w3), ch(Cji,w3), ch(Cij,w4), ch(Cjk,w5), ch(Cik,w5), ch(Cji,w5)]
   [@(w0)]
   [v(p,w0), v(¬p,w1), v(p,w2), v(¬p,w3), v(p,w4), v(¬p,w5)]
 ]

input >>> [com (j-=>i_Cij)][add_ch (i-j-k)][per (i-=>k_¬p)][del_ch(Cij)]((□<i_◇<i_p>>)∨¬Cij),w0
t0= □<i_◇<i_p>>
t1= t0∨¬Cij
world : w0
input : [t1]
announce_list =[com(j-=>i_Cij), add_ch(i-j-k), per(i-=>k_¬p), del_ch(Cij)]
now = w0
— channel check —
 ok.

— Com_i cij —

Agent_i link=[[w0,w1],[w0,w2]]

delete link for [cij].
Agent_i link=[[w0,w1]]
now = w0
— connect channel check —
Cij >>>> exist.
Cjk >>>> exist.
ok.

— add channel —
add channel cik.
now = w0
— channel check —
 ok.

— Per_k ~p —
Agent_k link=[[w0,w2],[w0,w4]]

insert link for [~p].
Agent_k link=[[w0,w2],[w0,w4],[w0,w1],[w0,w3],[w0,w5]]
now = w0

 channel list (w0)
[cij,cji,cjk,cik]

— del channel cij —

 channel list (w0)
[cji,cjk,cik]
```

図 4.24 $\mathfrak{M}_2, w_0 \models [\mathsf{com}^{\downarrow_i^j}\mathsf{C}_{ij}][\mathsf{C}_j^+(i,k)][\mathsf{per}^{\downarrow_k^i}\neg p][\mathsf{C}^-(i,j)]((\Box_i\Diamond_i p) \vee (\neg\mathsf{C}_{ij}))$ の入力結果 1

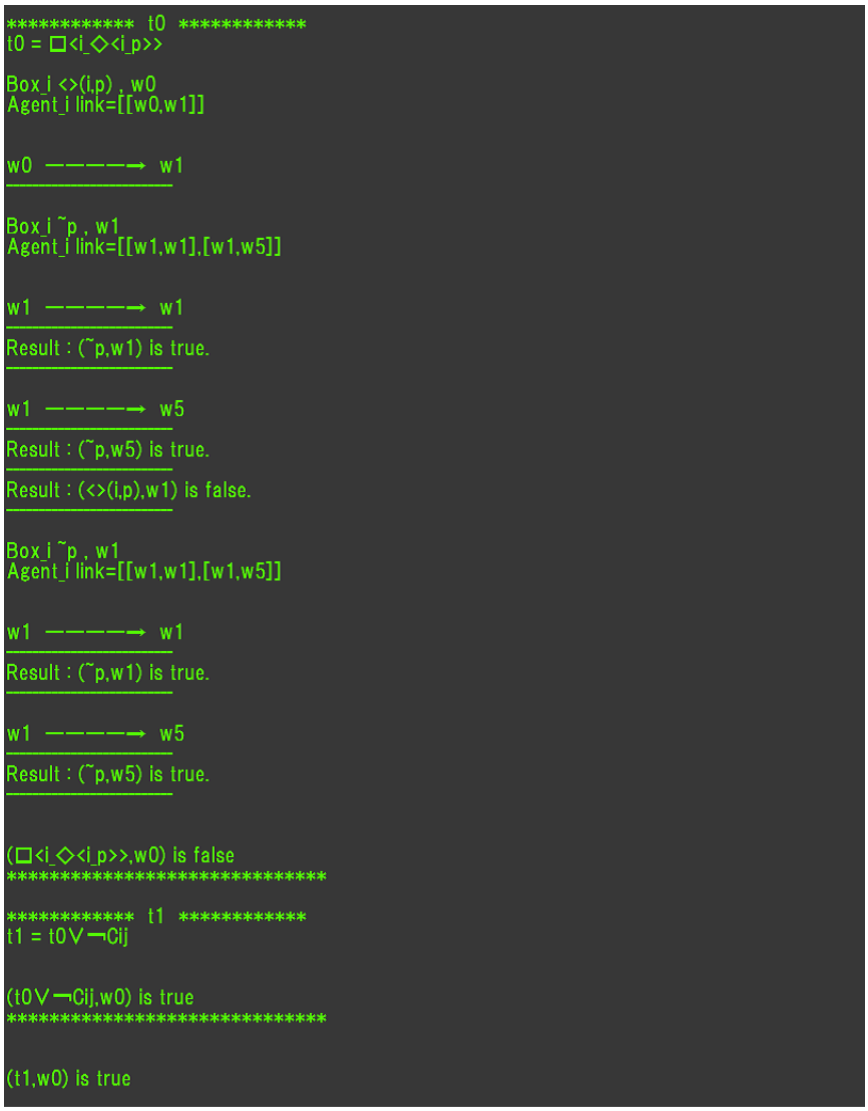


図 4.25 $\mathfrak{M}_2, w_0 \models [\mathsf{com}^{\downarrow_i^j}\mathsf{C}_{ij}][\mathsf{C}_j^+(i,k)][\mathsf{per}^{\downarrow_k^i}\neg p][\mathsf{C}^-(i,j)]((\Box_i\Diamond_i p) \vee (\neg\mathsf{C}_{ij}))$ の入力結果 2

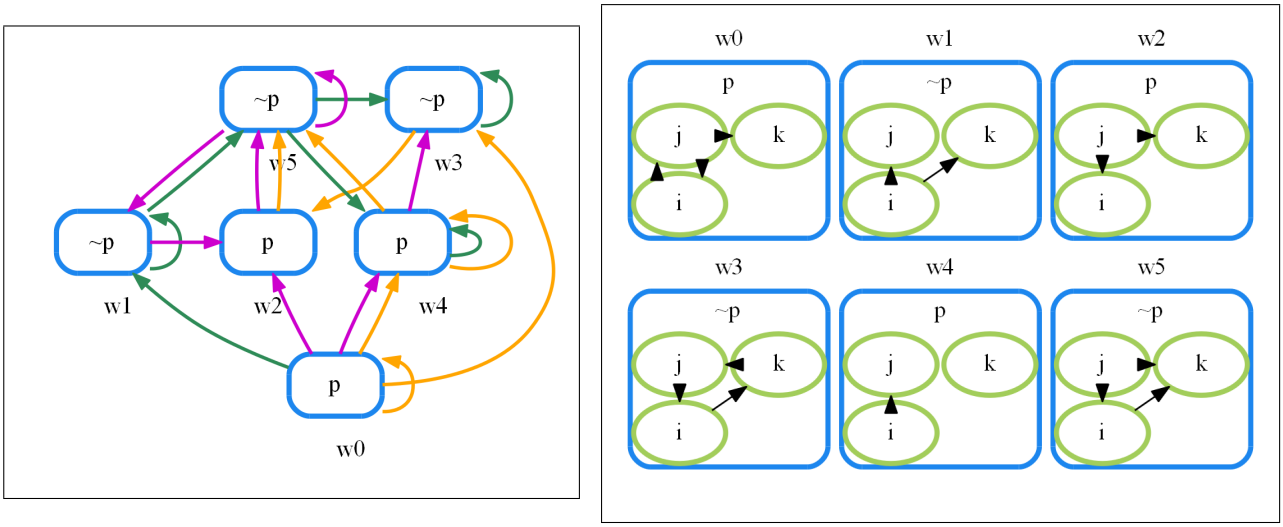


図 4.26 モデル $\mathfrak{M}_3$ の画像とチャネル関係図

図 4.26 では告知 $[\mathtt{com}^{\downarrow j}_{i}\mathtt{C}_{ij}]$ によりモデル画像で $w0$ から $w2$ へのアクセス関係が削除されている.

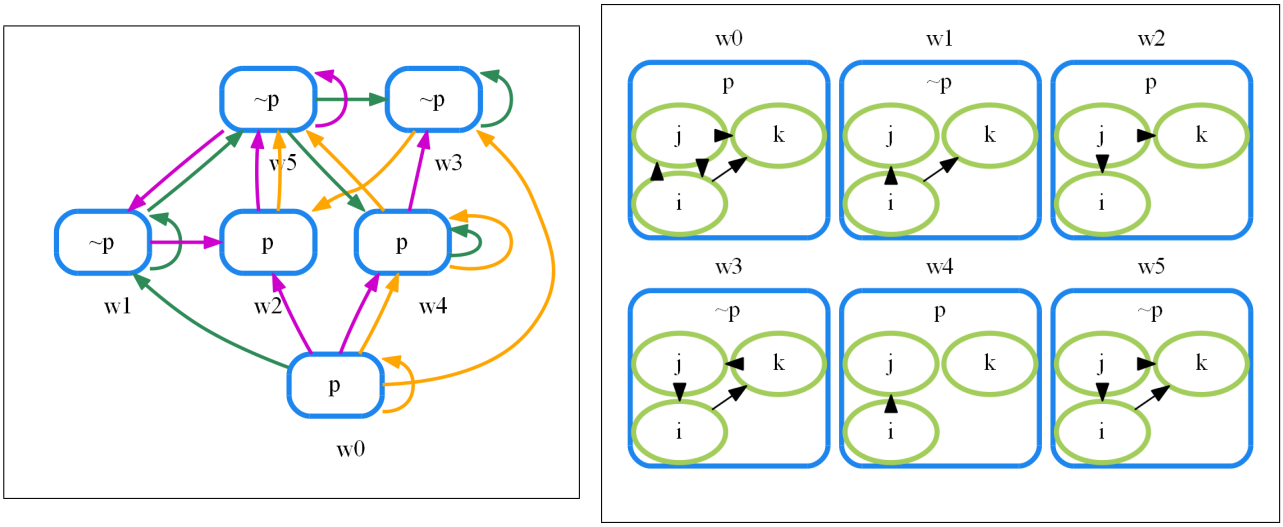


図 4.27 モデル $\mathfrak{M}_4$ の画像とチャネル関係図

図 4.27 では告知 $[\mathtt{C}^{+}_{j}(i,k)]$ によりチャネル関係図の世界 $w0$ でチャネル C_{ik} が追加されている.

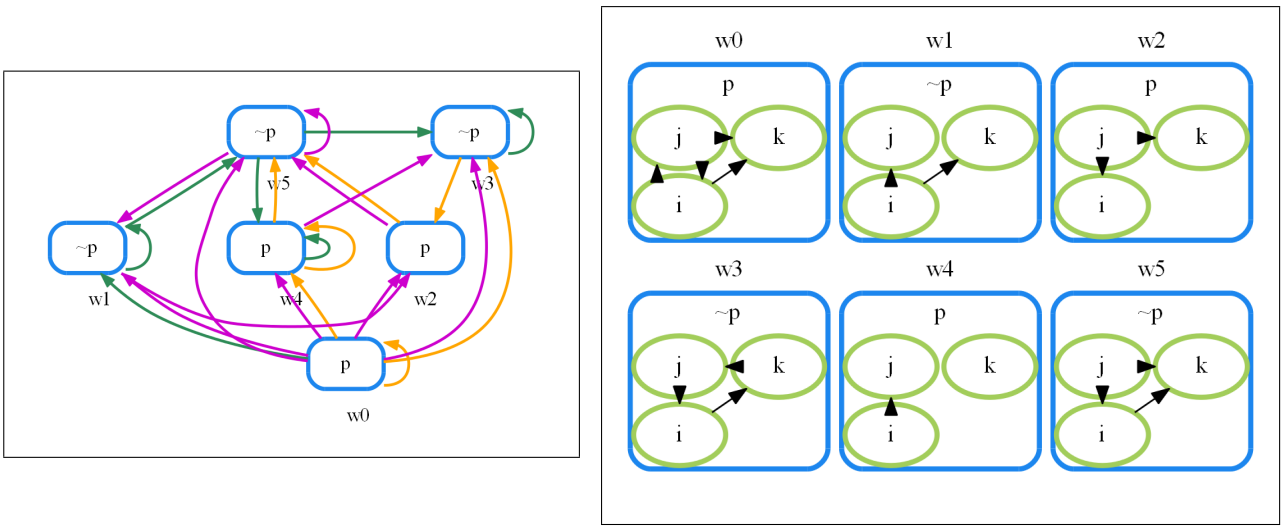


図 4.28 モデル $\mathfrak{M}_5$ の画像とチャネル関係図

図 4.28 では告知 $[\mathtt{per}^{\downarrow_k^i}\neg p]$ によりモデル画像でw0から$\neg p$ が成り立つ世界へのアクセス関係が追加されている.

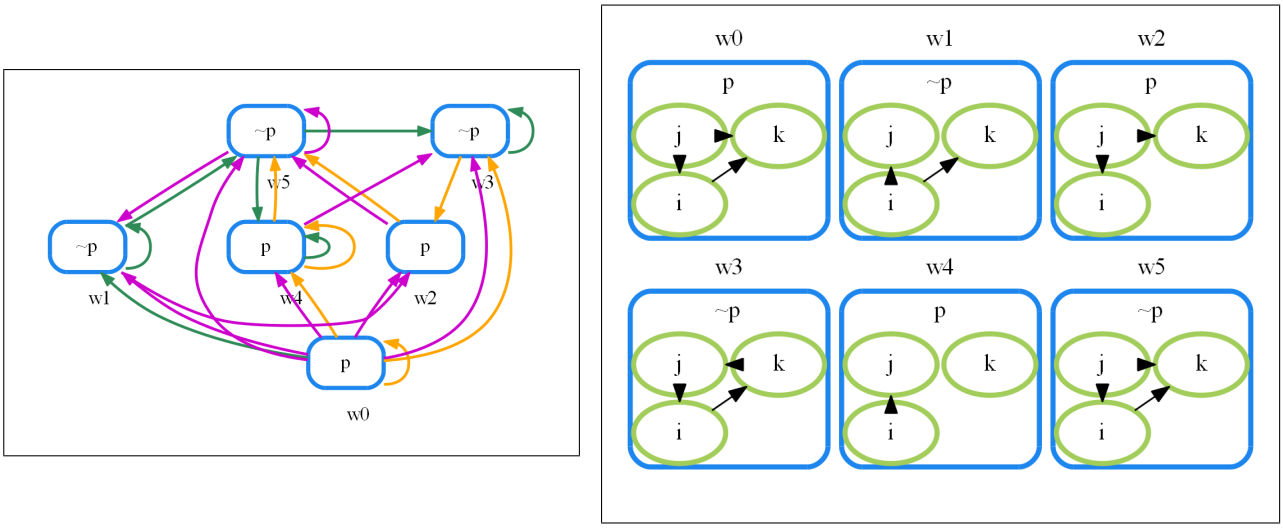


図 4.29 モデル $\mathfrak{M}_6$ の画像とチャネル関係図

図 4.29 では告知 $[\mathtt{C}^-(i,j)]$ によりチャネル関係図の $w0$ でチャネル $\mathtt{C}_{ji}$ が削除されている.

図 4.24 と図 4.25 より入力された告知について適切に処理をし, アクセス関係やチャネルの追加・削除をしているのが確認できる. 図 4.27 から図 4.29 を見るとそれぞれ告知の影響を受けモデルを更新しているのが分かり, $\mathfrak{M}_3$ では C_{ij} が成り立つ世界以外へのアクセス関係を削除し, $\mathfrak{M}_4$ ではエージェント i はエージェント k を介してエージェント j とチャネルを繋ぎ, $\mathfrak{M}_5$ では $\neg p$ が成り立つ世界へのアクセス関係を追加し, $\mathfrak{M}_6$ ではエージェント j からエージェント i へのチャネルを削除を行っている.

第5章　まとめ

5.1　考察

本研究では, エージェント間の通信のチャネルの変化に対応を実現させるために, Pimolluck ら [6] の commitment 告知と permission 告知について述べ, 次に小林 [4] らが提案したチャネル通信を形式化した. そして佐野ら [3] による commitment 告知のチャネル通信への対応や Seligman ら [8] の Facebook Logic の考え方を元に, permission 告知をエージェント間の通信チャネルによる告知に対応させ, さらに告知によるチャネルの追加・削除について形式化を行った. その後に形式化した論理式をを用いて計算機上に Java をベースとした論理システムの実装を行った. システムではシングルエージェントで様相論理, commitment 告知, permission 告知を検証し, その後にマルチエージェントでチャネル通信に対応した commitment 告知と permission 告知とチャネルの追加, 削除の告知を検証して, それらの動作を確認した. またシステムでは入力した論理式がモデルに対してどのような影響があるかを表現するためにクリプキモデルの画像をチャネル関係図を生成できるように実装を行った. 結果として告知を含んだ論理の計算と告知によってクリプキモデルやエージェント間のチャネル関係にがどのような影響を受けているかを確認することができた.

5.2　今後の課題

今後の課題について以下のものがある.

- 本研究では commitment 告知と permission 告知を通信チャネルを扱えるように形式化し, さらに告知によるチャネルの変更についての論理も形式化を行った. しかし, 公開告知論理のように公理系については定義しておらず, 形式化した論理式の妥当性を示すために公理系を定義する必要がある.
- 今回実装したシステムは問い合わせ通信を扱うことができず, 問い合わせ通信を含んだ論理に対応できるように改良する必要がある. 本システムでは問い合わせるエージェントを指定しており, その指定したエージェント経由で間接的なチャネル通信を行っているが, 問い合わせ通信を含んだ論理に対応することによってそのエージェント自身が介するエージェントを探し出し通信を行うことが可能となる. また問い

合わせ通信を行うことで告知を送る前に事前に他のエージェントの情報を知ることができ, 告知の失敗を未然に防げることも可能となる.

謝辞

本研究を進めるにあたり, 日頃から親切丁寧にご指導下さいました東条敏教授に深く感謝いたします. 佐野勝彦助教には研究全般にわたり多くの助言を頂き大変励みになりました. プログラミングに不慣れな私に適切な助言や励ましの言葉をかけて下さった先輩の秦野亮氏と野村尚新氏には感謝しきれません. 最後に共に励み, 時には競い合い, お互いを高め合った研究室の方々に深く感謝をします.

参考文献

[1] Johan van Benthem, Stefan Minica. Toward a Dynamic Logic of Questions, *Proceedings of Second International Workshop on Logic, Rationality and Interaction* (LORI-II), LNCS, vol. 5834, Springer-Verlag, pp. 27-41, 2009

[2] Foundation for Intelligent Physical Agents (FIPA). Communicative act library specification, `http://www.fipa.org/specs/fipa00037/`, 2002

[3] Katsuhiko Sano, Ryo Hatano, Satoshi Tojo, Misconception in legal cases from dynamic logical viewpoints, *Proceedings of the Sixth International Workshop of Juris − Informatics*, pp101–113, 2012.

[4] 小林幹門, 東条敏, エージェント間通信における信念の動的な更新, 人工知能学会誌, Vol. 24, No. 3, pp314–321, 2009.

[5] 沼岡千里, 大沢英一, 長尾確, マルチエージェントシステム, 協立出版, 1998.

[6] Pimolluck Jirakunkanok, Shinya Hirose, Katsuhiko Sano, Satoshi Tojo, Belief Rerevison in Chivalry Case, *Proceedings of the Seventh International Workshop on Juris − informatics*, 2013.

[7] Plaza, J. A., Logics of public communications In: M. L. Emrich, M. S. Pfeifer, M. Hadzikadic, and Z. W. Ras (eds.): *Proceedings of the 4th International Symposium on Methodologies for Intelligent Systems. pp.* 201 − 216.

[8] Jeremy Seligman, Fenrong Liu, Patrick Girard, Logic in the Community, *Proc. 4th Indian Conference on Logic and ITs Applicatiojns*, Lecture Notes in Computer Science, Vol.6521, pp178–188, 2011.

[9] 高玉圭樹, マルチエージェント学習, コロナ社, 2003.

[10] 東条敏, 言語・知識・信念の論理, オーム社, 2006.